CIRRELT

Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport

Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation

# Multi-Zone Multi-Trip Pickup and Delivery Problem with Time Windows and Synchronization

Phuong Khanh Nguyen
Teodor Gabriel Crainic
Michel Toulouse

March 2014

CIRRELT-2014-18

**Bureaux de Montréal :**
Université de Montréal
Pavillon André-Aisenstadt
C.P. 6128, succursale Centre-ville
Montréal (Québec)
Canada H3C 3J7
Téléphone : 514 343-7575
Télécopie : 514 343-7121

**Bureaux de Québec :**
Université Laval
Pavillon Palasis-Prince
2325, de la Terrasse, bureau 2642
Québec (Québec)
Canada G1V 0A6
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

Université Laval · McGill · Concordia Université · ÉTS · UQÀM · HEC Montréal · Polytechnique Montréal · Université de Montréal

# Multi-Zone Multi-Trip Pickup and Delivery Problem with Time Windows and Synchronization

**Phuong Khanh Nguyen**[1,2,*], **Teodor Gabriel Crainic**[1,3], **Michel Toulouse**[1,4]

[1] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT)

[2] Department of Computer Science and Operations Research, Université de Montréal, P.O. Box 6128, Station Centre-Ville, Montréal, Canada H3C 3J7

[3] Department of Management and Technology, Université du Québec à Montréal, P.O. Box 8888, Station Centre-Ville, Montréal, Canada H3C 3P8

[4] Department of Computer Science, Oklahoma State University, 700 North Greenwood Avenue, North Hall 328, Tulsa, OK 74106-0700, USA

**Abstract.** In this paper, we consider two-tier City Logistics systems accounting for both the inbound and outbound traffic that have not been taken into account in models and algorithms for vehicle routing research. The problem under study, called the Multi-zone Multi-trip Pickup and Delivery Problem with Time Windows and Synchronization, has two sets of intertwined decisions: the routing decisions which determine the sequence of customers visited by each vehicle route, the scheduling decisions which plan movements of vehicles between facilities within time synchronization restrictions. We propose a tabu search algorithm integrating multiple neighborhoods targeted to the decision sets of the problem. To assess the proposed algorithm, tests have been conducted on the first benchmark instances of the problem which have up to 72 facilities and 7200 customer demands. As no previous results are available in the literature for the problem, we also evaluate the performance of the method through comparisons with currently published results on the vehicle routing problem with backhauls. The proposed algorithm is competitive with other meta-heuristics both with and without time windows.

**Keywords**: Multi-trip pickup and delivery problem with time windows, synchronization, tabu search, multiple neighborhoods.

**Acknowledgements.** Partial funding for this project has been provided by the Natural Sciences and Engineering Research Council of Canada (NSERC), through its Industrial Research Chair and Discovery Grant programs, by the Université de Montréal, and by the EMME/2-STAN Royalty Research Funds (Dr. Heinz Spiess). We also gratefully acknowledge the support of Fonds de recherche du Québec through their infrastructure grants and of Calcul Québec and Compute Canada through access to their high-pefrformance computing infrastructure.

Results and views expressed in this publication are the sole responsibility of the authors and do not necessarily reflect those of CIRRELT.

Les résultats et opinions contenus dans cette publication ne reflètent pas nécessairement la position du CIRRELT et n'engagent pas sa responsabilité.

* Corresponding author: Phuong.NguyenKhanh@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2014

© Nguyen Khanh, Crainic, Toulouse and CIRRELT, 2014

## 1 Introduction

Research for city logistics is receiving more and more attention, with most researchers focusing on ***inbound*** freight flow. Hence, either the single-tier case or the multiple tiers case, freight flow is often considered in the direction from regions outside the city, say ***external zones***, to the city center, and the objective is to minimize the total cost of the associated system. In reality, freight is moved in, out, and through a city. When dedicated fleet is used for each type of freight flow, this might be simple to implement and manage, but it results in the presence of more vehicles and empty trips on the streets of the city, that eventually increases the operating costs and environmental pollution. In this paper, we therefore make an effort on integrating the ***outbound*** freight flow, shipping freight from the city center to external zones, with the 'classical' inbound freight flow into a single City Logistics system. We study a new problem, the so-called *Multi-zone Multi-trip Pickup and Delivery Problem with Time Windows and Synchronization*(MZT-PDTWS), which addresses an integrated system servicing both traffic types while sharing the same fleet of vehicles.

The MZT-PDTWS originates from planning the operations of two-tiered City Logistics systems (Crainic et al., 2009). In such systems, inbound loads are sorted and consolidated at first-tier facilities located on the outskirts of the city, moved to second-tier facilities, called satellites, located close to or within the city-center area, by a fleet of medium-sized vehicles. In the second tier, a smaller-capacity fleet performs tours to pick up outbound demands within the city center and transport them to satellites. Once at satellites, planned appropriate pairs of first-tier and second-tier vehicles transfer inbound and outbound demands to each other according to cross-docking, without intermediate storage. The first-tier vehicles then move the outbound demands to external zones, while the second-tier vehicles deliver the inbound demands to designated customers situated within the city center. This integration of inbound and outbound operations helps to reduce the number of empty vehicle movements in both tiers, as well as freight traffic in the city center. Since satellites are used as intermediate transshipment points for the freight distribution, the synchronization of the operations of first-tier and second-tier vehicles at satellites becomes one of the most constraining aspects of the problem.

In the MZT-PDTWS setting, a homogeneous fleet of vehicles operates out of a single garage to pickup- or delivery-customer demands associated to a given set of satellites. Each ***customer demand*** is defined by its specific location, commodity volume, as well as a particular service time requirement. Customer demands are divided into two categories, pickup (*backhaul*) demands representing outbound demands, and delivery (*linehaul*) demands representing inbound demands. The arrival of first-tier vehicles at a given time period define the set of delivery-customer demands to be serviced, and the time required to unload and transfer the freight thus define the availability period during which second-tier vehicles must arrive at the satellite and load. Second-tier vehicles must therefore synchronize their arrivals at satellites with these availability periods for loading the planned freight. The integration of outbound freight flow concerns pickup demands, which should be collected by second-tier vehicles, and brought to assigned satellites at their availability periods. Consequently, at satellites, second-tier vehicles may either unload pickup demands (if any), or load delivery demands (when available), or do both. In case the vehicle loads the planned freight, it undertakes a route delivering freight to the assigned customers. Otherwise, when the vehicle leaves a satellite empty (i.e., only unloads at this satellite), it either undergoes a route collecting new pickup demands, or moves empty to another satellite for loading delivery demands, to the garage to end its activity. The waiting stations may be used by vehicles to wait for its next appointment at a satellite. The MZT-PDTWS corresponds to the planning of the activities of second-tier vehicles.

The MZT-PDTWS is a generalization of the Time-dependent Multi-zone Multitrip Vehicle Routing Problem with Time Windows (TMZT-VRPTW) which has been studied in Crainic et al. (2012b) and Nguyen et al. (2013). The TMZT-VRPTW considers only the inbound traffic flow, while the MZT-PDTWS generalizes the TMZT-VRPTW by considering an additional traffic flow, say outbound, shipping freight in the opposite direction, i.e., from the city center to destinations outside the city limits. Both traffic types share a same fleet of vehicles. Furthermore, satellites would also be shared. This version has not been investigated in the literature. Such services sharing services increases the synchronization challenges at satellites, the complexity to manage the traffic of vehicles into and out of satellites in particular, as well as to route vehicles doing both pickup and delivery operations through satellites. We make three contributions: 1) we present a first formulation for the MZT-PDTWS, 2) we propose a first tabu search meta-heuristic to solve the problem, 3) a new benchmark with instances up to 72 supply points and 7200 customer demands is built to show the performance of the proposed method.

The remainder of the paper is organized as follows. Section 2 contains a detailed problem description. Section 3 reviews the literature. The problem formulation is then provided in Section 4. Details of the proposed methodology are described in Section 5. Computational results are then reported and analyzed in Section 6, while conclusions and future works are considered in Section 7.

## 2 Problem Description

In this paper, we address the integration of outbound traffic into city logistics. The physical flow from customers to satellites is called the pickup phase. The process from satellites to customers is called the delivery phase. Vehicles operate according to the ***Pseudo-Backhaul*** strategy described in Crainic et al. (2012a), in which any delivery or pickup phase must be completed before another one may be started. Each satellite may operate at several periods during the planning period considered. For the sake of simplicity, we define ***supply points*** as particular combinations of satellites and availability time periods in our model.

The MZT-PDTWS can be described as follows. There is a garage, or main depot, $g$, a set of pickup-customer demands $p \in \mathcal{C}^P$, a set of delivery-customer demands $d \in \mathcal{C}^D$, a set of waiting stations $w \in \mathcal{W}$, and a set of supply points $s \in \mathcal{S}$. A traveling cost (or travel time) $c_{i,j}$ is associated with each pair of $i$ and $j$ where $i, j \in g \cup \mathcal{C}^D \cup \mathcal{C}^P \cup \mathcal{S} \cup \mathcal{W}$. We use the terms cost, travel time, and distance interchangeably. Each supply point $s \in \mathcal{S}$ has a no-wait, hard opening time window $[t(s) - \eta, t(s)]$, specifying the earliest and latest times the vehicle may be at $s$, respectively. Hence, the vehicle must not arrive at $s$ sooner than $(t(s) - \eta)$ and no later than $t(s)$; in the former case, the vehicle has to wait at a waiting station $w \in \mathcal{W}$ before moving to $s$. Each customer demand is serviced by exactly one supply point. The supply point that services each delivery-customer demand is fixed and known in advance. For each pickup-customer demand $p \in \mathcal{C}^P$, it is given a set of supply points $\mathcal{S}_p \in \mathcal{S}$ that can service $p$. Therefore, it is required to assign each pickup-customer demand $p$ to exactly one supply point $s \in \mathcal{S}_p$. Consequently, each supply point $s$ may service a group of either pickup-customer demands $\mathcal{C}^P_s \subseteq \mathcal{C}^P$, or delivery-customer demands $\mathcal{C}^D_s \subseteq \mathcal{C}^D$ or both. Thus, it refers to movements where all freight collected from pickup-customer demands in $\mathcal{C}^P_s$ have to be transported to $s$ and all freight delivered to delivery-customer demands in $\mathcal{C}^D_s$ have to be loaded at $s$. Accordingly, each supply point $s$ requires an unloading time $\varphi'(s)$, which is the time required to unload freight picked up at a set of customer demands in $\mathcal{C}^P_s$, and a loading time $\varphi(s)$, which is the time required to load freight to service a set of customer demands in $\mathcal{C}^D_s$. For each customer demand $i \in \mathcal{C}^P \cup \mathcal{C}^D$, we use $(i, q_i, \delta(i), [e_i, l_i])$ to mean that customer demand $i$ requires a quantity $q_i$ of demand in the hard time window $[e_i, l_i]$ with a service time $\delta(i)$.

In general, once arriving at a supply point, the vehicle in the MZT-PDTWS may either unload pickup demands or load delivery demands or do both. Figure 1 represents these activities of a vehicle at a supply point $s$. The dashed lines stand for the empty move.

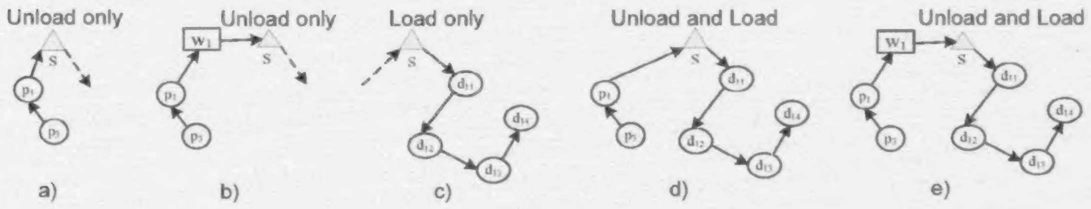


Figure 1: Activities at supply points

Figure 1a and 1b depict instances of 'unload only' operation in which after arriving at the supply point with the collected freight from pickup-customer demands, the vehicle unloads all freight, then it leaves the supply point empty for its next trip or goes to the depot to end its activity. From the last serviced pickup-customer demand, the vehicle may go directly to the supply point $s$ as shown in Figure 1a if it can arrive at $s$ within the time window $[t(s) - \eta, t(s)]$. Otherwise, in case the direct move gets the vehicle to $s$ sooner than $t(s) - \eta$, as shown in Figure 1b, the vehicle goes to a waiting station, and wait there in order to get to $s$ within the time window of $s$. Figure 1c represents the case of 'load only' operation in which the vehicle goes empty to supply point $s$ and loads freight. Figure 1d and 1e depict instances of 'unload and load' operation in which at a supply point $s$, after unloading all freight collected from pickup-customer demands, the vehicle starts to load freight, it then leaves $s$ to deliver freight to designated delivery-customer demands.

Let a pickup (delivery) leg be a route that links one or several pickup- (delivery-) customer demands of the same type with a supply point. Thus, we define two types for each pickup and delivery legs as well as their feasibility rules as follows:

- ***Direct-pickup leg*** is a route run by a vehicle that goes to one or several pickup-customer demands to collect freight and goes to the supply point directly to unload all freight (see Figure 1a).
- ***Indirect-pickup leg*** is similar to the case of direct-pickup leg, except that after servicing the last pickup-customer demand, the vehicle has to go to a waiting station and wait there due to the synchronization requirement at the supply point, then ends at the supply point to unload all freight (see Figure 1b).

  A pickup leg assigned to the supply point $s$ is feasible if the vehicle with a total load not exceeding $Q$ can arrive at $s$ within its time window $[t(s) - \eta, t(s)]$ after servicing a subset of pickup-customer demands in $\mathcal{C}_s^P$ within their time windows.
- ***Single-delivery leg*** is a route run by a vehicle that arrives empty at a supply point $s$ to load freight and delivers all freight to one or several delivery-customer demands in $\mathcal{C}_s^D$ (see Figure 1c). A single-delivery leg assigned to the supply point $s$ is feasible if the vehicle arrives empty at $s$ at time $t' \in [t(s) - \eta, t(s)]$ to load freight with a total load not exceeding $Q$, then leaves $s$ at time $t' + \varphi(s)$ to perform the delivery for serving a subset of customer demands in $\mathcal{C}_s^D$ within their time windows.
- ***Coordinate-delivery leg*** is a route run by a vehicle that starts empty at a supply point $s$ for loading freight, then delivers all freight to designated delivery-customer demands in $\mathcal{C}_s^D$, given that this vehicle does both unload and load at supply point $s$, i.e. it does a direct-pickup leg (see Figure 1d) or an indirect-pickup leg (see Figure 1e) at the same supply point $s$ right before this coordinate-delivery leg. A coordinate-delivery leg assigned to the supply point $s$ is feasible if the vehicle arrives at $s$ at time $t' \in [t(s) - \eta, t(s)]$ to unload all collected freight, it then starts to load delivery demands at time $t' + \varphi'(s)$ with a total load not exceeding $Q$, leaves $s$ at time $t(s) + \varphi'(s) + \varphi(s)$ to perform the delivery for serving a subset of customer demands in $\mathcal{C}_s^D$ within their time windows.

A sequence of legs, starting and ending at the main depot, assigned to a vehicle is called a ***work assignment***. For the sake of simplicity, from now on, the terms *vehicle* and *work assignment* are used interchangeably. Figure 2 illustrates a four-leg work assignment, where $s_1, s_2, s_3$ are supply points, $g$ and $w_1$ are respectively the main depot and waiting station, $\mathcal{C}^P_{s_1} = \{p_1, p_2, p_3, p_4, p_5\}$, $\mathcal{C}^D_{s_1} = \{d_1, d_2, d_3, d_4, d_5\}$, $\mathcal{C}^P_{s_2} = \{p_6, p_7, p_8, p_9, p_{10}\}$, $\mathcal{C}^D_{s_2} = \{d_6, d_7, d_8, d_9, d_{10}, d_{11}\}$, $\mathcal{C}^P_{s_3} = \{p_{11}, p_{12}, p_{13}, p_{14}, p_{15}\}$, and $\mathcal{C}^D_{s_3} = \{d_{12}, d_{13}, d_{14}, d_{15}\}$. The dashed lines stand for the empty arrival. This vehicle performs a sequence of four legs $\{r_1, r_2, r_3, r_4\}$ where $r_1 = \{s_1, d_1, d_3, d_4\}$ is a single-delivery leg, $r_2 = \{p_6, p_8, p_9, w_1, s_2\}$ is an indirect-pickup leg, $r_3 = \{s_2, d_6, d_9, d_8, d_7\}$ is a coordinate-delivery leg, and $r_4 = \{p_{11}, p_{13}, p_{12}, s_3\}$ is a direct-pickup leg. This vehicle first moves empty from the depot $g$ to supply point $s_1$. Once at $s_1$, this vehicle start loading delivery demands. After loading for a time $\varphi(s_1)$, it leaves $s_1$ to service customer demands $(d_1, d_3, d_4)$ in $\mathcal{C}^D_{s_1}$, then moves empty to pickup customer zone $\mathcal{C}^P_{s_2}$ for collecting freight at pickup-customer demands ($p_6$, $p_8$, $p_9$). In order to arrive at $s_2$ within its opening time window, after collecting freight from customer demand $p_9$, this vehicle has to go to waiting station $w_1$ and wait there. Once at $s_2$ (assuming arrival time is $t$), it does both unloading and loading operations: (1) at time $t$, it starts unloading and keeps doing for a time $\varphi'(s_2)$, and (2) then from time $t + \varphi'(s_2)$, it loads delivery demands and continues loading for a time $\varphi(s_2)$, after which it leaves $s_2$ to service customer demands ($d_6$, $d_9$, $d_8$, $d_7$) in $\mathcal{C}^D_{s_2}$. After servicing the last customer demand $d_7$, it moves empty to pickup customer zone $\mathcal{C}^P_{s_3}$. There, after loading freight at pickup-customer demands ($p_{11}, p_{13}, p_{12}$), this vehicle moves to supply point $s_3$ within the opening time window of $s_3$. Once at $s_3$, this vehicle starts unloading freight for a duration of $\varphi'(s_3)$. At the end, this vehicle moves empty back to the depot $g$ to complete its task.

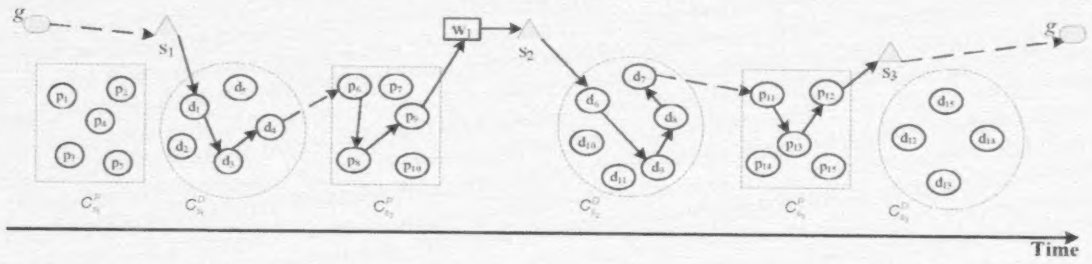


Figure 2: A four-leg work assignment illustration

The MZT-PDTWS can be seen as the problem of (1) assigning pickup-customer demands to supply points, and (2) determining a set of pickup and delivery legs and an assignment of each leg to one vehicle, such that each vehicle can perform several legs sequentially. The objective is to minimize the total cost, which is comprised of routing cost and fixed cost on the use of vehicles, while the following conditions are satisfied:

1. Every vehicle starts and ends its leg sequence at the main depot $g$;

2. Each pickup-customer demand $p$ is assigned to exactly one supply point $s \in \mathcal{S}_p$;

3. Every vehicle required to service customer demands in $\mathcal{C}_s^P \cup \mathcal{C}_s^D$ must reach its supply point $s \in \mathcal{S}$ within its no-wait, hard opening time window. Assume the arrival time at $s$ is $t$; Once at $s$:

   - if the vehicle is not empty, i.e., it is containing goods picked up from customer demands in $\mathcal{C}_s^P$, it has to unload them first. The vehicle starts unloading goods at time $t$, and continues unloading for a duration of $\varphi'(s)$, after which it may either:
     - (1) load goods shipped from external zones for a duration of $\varphi(s)$ and then leave $s$ to deliver goods to customer demands in $\mathcal{C}_s^D$, or
     - (2) move empty either to another pickup customer zone to collect goods, or directly to another supply point for loading goods, or
     - (3) go to the main depot $g$ to complete its task;
   - otherwise, the vehicle starts to load goods to service customer demands in $\mathcal{C}_s^D$. It starts loading goods at time $t$ and continues loading for a duration of $\varphi(s)$, after which it leaves $s$ to deliver goods to customers in $\mathcal{C}_s^D$. After performing a trip within the delivery customer zone $\mathcal{C}_s^D$, the vehicle may continue its movement as either the situation (2) or (3) described above;

4. Every customer demand is visited by exactly one leg with a total load not exceeding $Q$, and each customer demand $i \in \mathcal{C}^D \cup \mathcal{C}^P$ is serviced within its hard time window $[e_i, l_i]$, i.e., a vehicle may arrive before $e_i$ and wait to begin service, but must not arrive later than $l_i$.

## 3 Literature Review

In this paper, we present a new variant of the VRP. It extends the Time-dependent Multi-zone Multi-trip Vehicle Routing problem with Time Windows (TMZT-VRPTW) by considering an additional type of customer demands. The TMZT-VRPTW just addresses inbound demands, resulting in only one type of customers, i.e., delivery-customer demands, while the MZT-PDTWS considers delivery and pickup-customer demands as it addresses both inbound and outbound demands. Crainic et al. (2009) pioneered the introduction of the TMZT-VRPTW and proposed a decomposition-based heuristic approach to solve it. The general idea is to solve each customer-zone routing out of each supply point subproblem independently, and then put the created vehicle tours together into multi-tour routes by solving a minimum cost network flow problem. Yet, as routing decisions affect the supply point assignment decisions and vice versa, these two decision levels are intertwined and should not be solved separately. Nguyen et al. (2013) later investigated an alternative approach that addresses these two decisions simultaneously, in a tabu search framework. Thus, in comparing to the previous approach, it yields solutions with higher quality up to 4.42% in term of total cost, requiring not only less vehicles, but also less usage of waiting stations.

In the context of Pickup and Delivery problems, there has been extensive research on variants of the problem with respect to additional and different types of constraints which occur in real-world applications; see, e.g., a number of surveys (Savelsbergh and Solomon, 1995; Parragh et al., 2008a,b; Berbeglia et al., 2007, 2010) and book (Toth and Vigo, 2002). Based on the difference in transportation endpoints, Parragh et al. (2008a,b) divided them into two subclasses: the first refers to transportation of goods from the depot to delivery (*linehaul*) customers and from pickup (*backhaul*) customers to the depot; the second refers to those problems where goods are transported between pickup and delivery locations. As we follow the Pseudo-Backhaul strategy in which any delivery or pickup phase must be completed before another one may be started, our problem belongs to the first subclass.

In reviewing the literature, the first subclass of Pickup and Delivery problems includes the single demand case where linehaul and backhaul customers are disjoint, and the combined case where the same customer has both a pickup and a delivery demand. In the former case, there are either problems in which linehaul customers of a given trip have to be serviced before backhaul customers of the same trip (Osman and Wassan, 2002; Brandão, 2006), that are denoted as Vehicle Routing problem with Backhauls (VRPB); or problems in which linehaul and backhaul customers may be visited in any order (Dethloff, 2002; Ropke and Pisinger, 2006), that are denoted as Vehicle Routing problem with Mixed linehauls and Backhauls (VRPMB). In the combined case, each customer may be either visited exactly once (Nagy and Salhi, 2005; Dell'Amico et al., 2006) or visited twice, one for delivery and one for pickup (Salhi and Nagy, 1999; Gribkovskaia et al., 2001). Problems in this case are denoted as Vehicle Routing problem with Simultaneous Delivery and Pickup.

In our problem, a customer, which is identified by a location, may have both types of demands: pickup and delivery. These demands of a customer might be available at different periods with different commodity volumes, thus we define them as customer demands. However, pickup and delivery phases are completed separately. The VRPB can be considered as a subproblem of our problem. More precisely, VRPB can be seen as the problem of routing delivery-customer demands of supply point $s$ and pickup-customer demands assigned to supply point $s'$ where $t(s) < t(s')$ in our problem, while time synchronization restrictions at supply points and waiting stations are not considered. Starting from the supply point $s$, the vehicles first deliver freight to delivery-customer demands. Then, they collect new freight at pickup-customer demands and bring to supply point $s'$. Two variants with and without time windows at customers are considered in the VRPB literature. However, the number of studies dealing with the time window variant is relatively smaller than those without time window.

Vehicle Routing problem with Cross-Docking (VRPCD) is a VRP variant sharing synchronization of vehicles' operations requirement with our problem. In general, the VRPCD involves transporting products from a set of suppliers to their corresponding customers via a cross-dock. More precisely, products from the suppliers are picked up by a fleet of vehicles, consolidated at the cross-dock (i.e., classified into a certain group according to their destination), and immediately delivered to customers by the same set of vehicles, without delay or storage. A supplier and its corresponding customers are not necessarily served by the same vehicle. At the cross-dock, for each vehicle the unloading must be completed before reloading starts. There may exist constraint on simultaneous arrival at cross-dock for all the vehicles (Lee et al., 2006; Liao et al., 2010) or the arrival dependency among the vehicles is determined by the consolidation decisions (Wen et al., 2008). As our problem, each vehicle in the VRPCD operates two phases pickup and delivery separately. However, there are also differences between the VRPCD and the MZT-PDTWS.

In the VRPCD, each vehicle performs a sequence of two trips, i.e., pickup and then delivery, using cross-dock as intermediate storage. While in our problem, there are neither limitation on the number of trips nor the requirement for the ordering between pickup and delivery trips performing by each vehicle. Vehicles in our problem are synchronized at multiple locations (supply points) rather than a single location (the cross-dock) in the VRPCD. The synchronized arrival time of all or several vehicles at the cross-dock is considered as a variable in the VRPCD, which is determined by the consolidation decisions, while it is given in advance in our problem.

## 4 Model Formulation

### 4.1 Notation

We define for each supply point $s \in \mathcal{S}$:

- $A_s^{DS} = \{(d,s) | s' \in \mathcal{S}, d \in \mathcal{C}_{s'}^D, t(s') < t(s), e_d + \delta(d) + c_{d,s} \leq t(s) \ \}$ contains all the arcs $(d,s)$ from delivery-customer demands $d$ to the supply point $s$ such that $e_d + \delta(d) + c_{d,s} \leq t(s)$, i.e., the vehicle could arrive at $s$ before the opening time $t(s)$. By including this constraint, we eliminate inadmissible arcs, thus reduce the set;
- $A_s^{PS} = \{(p,s) | p \in \mathcal{C}_s^P\}$ contains all the arcs from pickup-customer demands $p$ to the supply point $s$ such that $s \in S_p$.
- $A_s^{S^-} = \{(s',s) | t(s) - \eta \leq t(s') - \eta + \varphi'(s') + c_{s',s} \leq t(s)\}$ contains all the arcs $(s',s)$ from any supply points $s'$ to the supply point $s$ such that $t(s) - \eta \leq t(s') - \eta +$

$\varphi'(s') + c_{s',s} \leq t(s)$, i.e., the vehicle could travel directly from $s'$ to $s$ when it only unloads at $s'$ (leaves $s'$ empty);

- $A_s^{S+} = \{(s,s')|t(s') - \eta \leq t(s) - \eta + \varphi'(s) + c_{s,s'} \leq t(s')\}$ contains all the arcs $(s,s')$ from the supply point $s$ to any supply points $s'$ such that $t(s') - \eta \leq t(s) - \eta + \varphi'(s) + c_{s,s'} \leq t(s')$, i.e., the vehicle could travel directly from $s$ to $s'$ when it only unloads at $s$;
- $A_s^{SP} = \{(s,p)|t(s) - \eta + \varphi'(s) + c_{s,p} \leq l_p, p \in \mathcal{C}^P\}$ contains all the arcs $(s,p)$ from the supply point $s$ to any pickup-customer demands $p \in \mathcal{C}^P$ such that $t(s) - \eta + \varphi'(s) + c_{s,p} \leq l_p$, i.e., the vehicle could arrive at $p$ from $s$ before the due time $l_p$ when it only unloads at $s$ (there exists arcs $(s,p)$, i.e., the vehicle goes from $s$ to $p$, only if the vehicle leaves $s$ empty, i.e., only unloads at $s$);
- $A_s^{SD} = \{(s,d)|d \in \mathcal{C}_s^D\}$ contains all the arcs from the supply point $s$ to any delivery-customer demands $d \in \mathcal{C}_s^D$;

We define for each delivery-customer demand $d \in \mathcal{C}_s^D, s \in \mathcal{S}$:

- $A_d^{DS} = \{(d,s')|s' \in \mathcal{S}, e_d + \delta(d) + c_{d,s'} \leq t(s')\}$ contains all the arcs $(d,s')$ from the delivery-customer demand $d$ to any supply points $s' \in \mathcal{S}$ such that $e_d + \delta(d) + c_{d,s'} \leq t(s')$, i.e., the vehicle could arrive at $s'$ from $d$ before the opening time $t(s')$;
- $A_d^{DP} = \{(d,p)|p \in \mathcal{C}^P, e_d + \delta(d) + c_{d,p} \leq l_p\}$ contains all the arcs $(d,p)$ from the delivery-customer demand $d$ to any pickup-customer demands $p \in \mathcal{C}^P$ such that $e_d + \delta(d) + c_{d,p} \leq l_p$, i.e., the vehicle could arrive at $p$ before the due time $l_p$;
- $A_d^{D+} = \{(d,d')|d' \in \mathcal{C}_s^D, e_d + \delta(d) + c_{d,d'} \leq l_{d'}\}$ contains all the arcs $(d,d')$ from the delivery-customer demand $d$ to any other delivery-customer demands $d'$ of the same zone $\mathcal{C}_s^D$ such that $e_d + \delta(d) + c_{d,d'} \leq l_{d'}$;
- $A_d^{D-} = \{(d',d)|d' \in \mathcal{C}_s^D, e_{d'} + \delta(d') + c_{d',d} \leq l_d\}$ contains all the arcs $(d',d)$ from any delivery-customer demands $d'$ of the same zone $\mathcal{C}_s^D$ to the delivery-customer demand $d$ such that $e_{d'} + \delta(d') + c_{d',d} \leq l_d$;

We define for each pickup-customer demand $p \in \mathcal{C}^P$:

- $A_p^{PS} = \{(p,s)|s \in \mathcal{S}_p\}$ contains all the arcs from the pickup-customer demand $p$ to its available supply point $s \in \mathcal{S}_p$;
- $A_p^{PD} = \{(p,d)|d \in \mathcal{C}^D, e_p + \delta(p) + c_{p,d} \leq l_d\}$ contains all the arcs $(p,d)$ from the pickup-customer demand $p$ to any delivery-customer demands $d$ such that $e_p + \delta(p) + c_{p,d} \leq l_d$;

- $A_p^{P+} = \{(p,p')|p' \in \mathcal{C}^P, e_p + \delta(p) + c_{p,p'} \leq l_{p'}, S_p \cap S_{p'} \neq \emptyset\}$ contains all the arcs $(p,p')$ from the pickup-customer demand $p$ to any other pickup-customer demands $p'$ such that (1) $S_p \cap S_{p'} \neq \emptyset$, i.e., $p'$ has at least one available supply point in common with $p$, and (2) $e_p + \delta(p) + c_{p,p'} \leq l_{p'}$;
- $A_p^{P-} = \{(p',p)|p' \in \mathcal{C}^P, e_{p'} + \delta(p') + c_{p',p} \leq l_p, S_p \cap S_{p'} \neq \emptyset\}$ contains all the arcs $(p',p)$ from any pickup-customer demands $p'$ to the pickup-customer demand $p$ such that (1) $S_p \cap S_{p'} \neq \emptyset$, i.e., $p'$ has at least one available supply point in common with $p$, and (2) $e_{p'} + \delta(p') + c_{p',p} \leq l_p$;
- $A_p^{DP} = \{(d,p)|d \in \mathcal{C}^D, e_d + \delta(d) + c_{d,p} \leq l_p\}$ contains all the arcs $(d,p)$ from any delivery-customer demands $d$ to the pickup-customer demand $p$ such that $e_d + \delta(d) + c_{d,p} \leq l_p$;
- $A_p^{SP} = \{(s,p)|t(s) - \eta + \varphi'(s) + c_{s,p} \leq l_p\}$ contains all the arcs $(s,p)$ from any supply points $s$ to the pickup-customer demand $p$ such that $t(s) - \eta + \varphi'(s) + c_{s,p} \leq l_p$, i.e., the vehicle could arrive at $p$ from $s$ before the due time $l_p$ when it only unload at $s$;

We define for each waiting station $w \in \mathcal{W}$:

- $A^{W-} = \{(i,w)|i \in \{\mathcal{S} \cup \mathcal{C}^D \cup \mathcal{C}^P\}, w \in \mathcal{W}\}$ contains all the arcs from pickup-customer demands, delivery-customer demands and supply points to waiting stations;
- $A^{WS} = \{(w,s)|w \in \mathcal{W}, s \in \mathcal{S}\}$ contains all the arcs from waiting stations to supply points;

For the depot $g$, we define:

- $A^{DG} = \{(d,g)|d \in \mathcal{C}^D\}$ contains all the arcs from delivery-customer demands $d$ to the main depot $g$;
- $A^{GS} = \{(g,s)|s \in \mathcal{S}\}$ contains all the arcs from the main depot $g$ to supply points;
- $A^{GP} = \{(g,p)|p \in \mathcal{C}^P\}$ contains all the arcs from the main depot $g$ to pickup-customer demands $p$;

Let $F$ stand for fixed cost for operating a vehicle. The set of available vehicles is denoted by $\mathcal{K}$. The maximal number of arcs included in any vehicle is given by $e$ and we define $\mathcal{R}$ as $\{1,...,e\}$. Define by $M$ a large positive constant.

## 4.2 Formulation

The MZT-PDTWS is defined on a space-time network $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, where $\mathcal{V}$ is the set of nodes, and the arcs in $\mathcal{A}$ stand for the possible movements between these nodes. Set $\mathcal{V}$ is made up of the main depot $g$ and the sets of customer demands, supply points and waiting stations, i.e., $\mathcal{V} = g \cup \mathcal{C}^P \cup \mathcal{C}^D \cup \mathcal{S} \cup \mathcal{W}$. Set $\mathcal{A} = \cup_{s \in \mathcal{S}}[A_s^{SD} \cup A_s^{SP} \cup A_s^{S^+}] \cup_{d \in \mathcal{C}^D} [A_d^{DS} \cup A_d^{DP} \cup A_d^{D^+}] \cup_{p \in \mathcal{C}^P} [A_s^{PS} \cup A_s^{PD} \cup A_s^{P^+}] \cup A^{W^-} \cup A^{WS} \cup A^{DG} \cup A^{GS} \cup A^{GP}$, which consists of admissible arcs.

We define the following decision variables:

- $x_{ijk}^r$, a binary variable that takes value 1 if arc $(i, j) \in \mathcal{A}$ is traversed by vehicle $k$ and appears in the $r$th position of the work assignment of vehicle $k$, and value 0 otherwise;
- $y_{ps}$, a binary variable that takes value 1 if pickup-customer demand $p \in \mathcal{C}^P$ is assigned to supply point $s \in \mathcal{S}$, and value 0 otherwise;
- $z_{sk}$, a binary variable that takes value 1 if vehicle $k$ unloads at supply point $s$, and value 0 otherwise.

Note that we preliminary set $y_{ps} = 0 \; \forall p \in \mathcal{C}^P, s \notin S_p$ given that such $s$ does not service $p$. Demands at each supply point $s \in \mathcal{S}$, waiting station $w \in \mathcal{W}$ and the main depot $g$ are equal to zero, i.e, $q_s = q_w = q_g = 0$. For convenience, we set demand at each delivery node $d \in \mathcal{C}^D$: $q_d = -q_d < 0$. In addition,

| | |
|---|---|
| $B_{ik}$ | is the starting time of service at customer demand $i \in \mathcal{C}^P \cup \mathcal{C}^D$ by vehicle $k$; |
| $B_{sk}$ | is the arrival time of vehicle $k$ at supply point $s \in \mathcal{S}$; |
| $B_{iwk}$ | is the arrival time of vehicle $k$ at waiting station $w \in \mathcal{W}$ from a supply point, a delivery-customer demand, or a pickup-customer demand $i \in \{\mathcal{S} \cup \mathcal{C}^D \cup \mathcal{C}^P\}$; |
| $Q_{ik}$ | is the load of vehicle $k$ when leaving $i \in \mathcal{V}$; |
| $L_{sk}$ | is the load of vehicle $k$ when arriving at supply point $s \in \mathcal{S}$. |

We set $Q_{gk} = 0 \; \forall k \in \mathcal{K}$ as the vehicle leaves the depot empty.

The MZT-PDTWS can then be formulated as the following:

$$\text{Minimize} \sum_{k \in \mathcal{K}} \sum_{(i,j) \in \mathcal{A}} c_{ij} \sum_{r \in \mathcal{R}} x_{ijk}^r + F \sum_{k \in \mathcal{K}} \left( \sum_{s \in \mathcal{S}} x_{gsk}^1 + \sum_{p \in \mathcal{C}^P} x_{gpk}^1 \right) \tag{1}$$

$$\text{S.t.} \sum_{r\in\mathcal{R}} \left( x^r_{sgk} + \sum_{(s,d)\in A_s^{SD}} x^r_{sdk} + \sum_{w\in\mathcal{W}} x^r_{swk} + \sum_{(s,s')\in A_s^{S+}} x^r_{ss'k} + \sum_{(s,p)\in A_s^{SP}} x^r_{spk} \right) \le 1 \quad \forall s \in \mathcal{S}, k \in \mathcal{K} \tag{2}$$

$$\begin{aligned} x^1_{gsk} + \sum_{r\in\mathcal{R}} \left( \sum_{w\in\mathcal{W}} x^r_{wsk} + \sum_{(s',s)\in A_s^{S-}} x^r_{s'sk} + \sum_{(p,s)\in A_s^{PS}} x^r_{psk} + \sum_{(d,s)\in A_s^{DS}} x^r_{dsk} \right) \\ = \sum_{r\in\mathcal{R}} \left( x^r_{sgk} + \sum_{(s,d)\in A_s^{SD}} x^r_{sdk} + \sum_{w\in\mathcal{W}} x^r_{swk} + \sum_{(s,s')\in A_s^{S+}} x^r_{ss'k} + \sum_{(s,p)\in A_s^{SP}} x^r_{spk} \right) \\ \forall s \in \mathcal{S}, k \in \mathcal{K} \end{aligned} \tag{3}$$

$$\sum_{s\in\mathcal{S}} x^1_{gsk} + \sum_{p\in\mathcal{C}^P} x^1_{gpk} = \sum_{r\in\mathcal{R}} \left( \sum_{d\in\mathcal{C}^D} x^r_{dgk} + \sum_{s\in\mathcal{S}} x^r_{sgk} \right) \quad \forall k \in \mathcal{K} \tag{4}$$

$$\sum_{r\in\mathcal{R}} \sum_{s\in\mathcal{S}} x^r_{wsk} = \sum_{r\in\mathcal{R}} \left( \sum_{d\in\mathcal{C}^D} x^r_{dwk} + \sum_{p\in\mathcal{C}^P} x^r_{pwk} + \sum_{s\in\mathcal{S}} x^r_{swk} \right) \quad \forall w \in \mathcal{W}, k \in \mathcal{K} \tag{5}$$

$$\sum_{s\in\mathcal{S}_p} y_{ps} = 1 \quad \forall p \in \mathcal{C}^P \tag{6}$$

$$\sum_{k\in\mathcal{K}} \sum_{r\in\mathcal{R}} x^r_{psk} \le y_{ps} \quad \forall p \in \mathcal{C}^P, s \in \mathcal{S} \tag{7}$$

$$x^r_{pwk} + y_{ps} \le x^{r+1}_{wsk} + 1 \quad \forall p \in \mathcal{C}^P, s \in \mathcal{S}_p, w \in \mathcal{W}, r \in \mathcal{R}, k \in \mathcal{K} \tag{8}$$

$$\sum_{k\in\mathcal{K}} \sum_{r\in\mathcal{R}} x^r_{pp'k} + y_{ps} \le y_{p's} + 1 \quad \forall p, p' \in \mathcal{C}^P, p \ne p', s \in \mathcal{S}_p \tag{9}$$

$$\sum_{k\in\mathcal{K}} \sum_{r\in\mathcal{R}} \left( \sum_{(p,p')\in A_p^{P+}} x^r_{pp'k} + \sum_{w\in\mathcal{W}} x^r_{pwk} + \sum_{s\in\mathcal{S}_p} x^r_{psk} \right) = 1 \quad \forall p \in \mathcal{C}^P \tag{10}$$

$$\sum_{k\in\mathcal{K}} \sum_{r\in\mathcal{R}} \left( \sum_{(s,p)\in A_p^{SP}} x^r_{spk} + \sum_{(p',p)\in A_p^{P-}} x^r_{p'pk} + \sum_{(d,p)\in A_p^{DP}} x^r_{dpk} \right) + \sum_{k\in\mathcal{K}} x^1_{gpk} = 1 \quad \forall p \in \mathcal{C}^P \tag{11}$$

$$\sum_{k\in\mathcal{K}}\sum_{r\in\mathcal{R}}\left(\sum_{(d,d')\in A_d^{D^+}} x^r_{dd'k} + \sum_{(d,p)\in A_d^{DP}} x^r_{dpk} + \sum_{(d,s')\in A_d^{DS}} x^r_{ds'k} + \sum_{w\in\mathcal{W}} x^r_{dwk} + x^r_{dgk}\right) = 1 \quad \forall d \in \mathcal{C}^D_s, s\in\mathcal{S} \tag{12}$$

$$\sum_{k\in\mathcal{K}}\sum_{r\in\mathcal{R}}\left(x^r_{sdk} + \sum_{(d',d)\in A_d^{D^-}} x^r_{d'dk}\right) = 1 \;\; \forall d\in\mathcal{C}^D_s, s\in\mathcal{S} \tag{13}$$

$$Q_{jk} \geq (Q_{ik} + q_j) - Q(1-\sum_{r\in\mathcal{R}} x^r_{ijk}) \;\; \forall (i,j)\in\mathcal{A}, j\notin\mathcal{S}, k\in\mathcal{K} \tag{14}$$

$$L_{sk} \geq Q_{ik} - Q(1-\sum_{r\in\mathcal{R}} x^r_{isk}) \; \forall (i,s)\in\mathcal{A}, s\in\mathcal{S}, k\in\mathcal{K} \tag{15}$$

$$max\{0, q_i\} \leq Q_{ik} \leq min\{Q, Q+q_i\} \;\; \forall i\in\mathcal{V}, k\in\mathcal{K} \tag{16}$$

$$Q_{sk} \leq Q \sum_{d\in\mathcal{C}^D_s}\sum_{r\in\mathcal{R}} x^r_{sdk} \;\; \forall s\in\mathcal{S}, k\in\mathcal{K} \tag{17}$$

$$Q_{sk} \geq \min_{d\in\mathcal{C}^D_s}\{q_d\} \sum_{d\in\mathcal{C}^D_s}\sum_{r\in\mathcal{R}} x^r_{sdk} \;\; \forall s\in\mathcal{S}, k\in\mathcal{K} \tag{18}$$

$$L_{sk} \geq \min_{p\in\mathcal{C}^P}\{q_p\} \sum_{i\in\mathcal{V}\backslash\mathcal{C}^D}\sum_{r\in\mathcal{R}} x^r_{sik} \;\; \forall s\in\mathcal{S}, k\in\mathcal{K} \tag{19}$$

$$Q_{sk} \leq Q(1-\sum_{i\in\mathcal{V}\backslash\mathcal{C}^D}\sum_{r\in\mathcal{R}} x^r_{sik}) \;\; \forall s\in\mathcal{S}, k\in\mathcal{K} \tag{20}$$

$$\sum_{k\in\mathcal{K}} L_{sk} = \sum_{p\in\mathcal{C}^P} q_p y_{ps} \;\; \forall s\in\mathcal{S} \tag{21}$$

$$\begin{gathered} B_{jk} \geq B_{ik} + \delta(i) + c_{i,j} - M(1-\sum_{r\in\mathcal{R}} x^r_{ijk}) \\ \forall (i,j)\in\mathcal{A}, i\in\{\mathcal{C}^P\cup\mathcal{C}^D\}, j\in\{\mathcal{C}^P\cup\mathcal{C}^D\cup\mathcal{S}\}, i\neq j, k\in\mathcal{K} \end{gathered} \tag{22}$$

$$B_{ik} \geq B_{sk} + \varphi'(s) + c_{s,i} - M(1-\sum_{r\in\mathcal{R}} x^r_{sik}) \;\; \forall s\in\mathcal{S}, i\in\{\mathcal{C}^P\cup\mathcal{S}\setminus s\}, k\in\mathcal{K} \tag{23}$$

$$B_{dk} \geq B_{sk} + \varphi(s) + z_{sk}\varphi'(s) + c_{s,d} - M(1 - \sum_{r \in \mathcal{R}} x^r_{sdk}) \quad \forall s \in \mathcal{S}, d \in \mathcal{C}^D_s, k \in \mathcal{K} \tag{24}$$

$$B_{iwk} \geq B_{iw} + \delta(i) + c_{i,w} - M(1 - \sum_{r \in \mathcal{R}} x^r_{iwk}) \quad \forall w \in \mathcal{W}, i \in \{\mathcal{C}^P \cup \mathcal{C}^D\}, k \in \mathcal{K} \tag{25}$$

$$B_{swk} \geq B_{sk} + \varphi'(s) + c_{s,w} - M(1 - \sum_{r \in \mathcal{R}} x^r_{swk}) \quad \forall w \in \mathcal{W}, s \in \mathcal{S}, k \in \mathcal{K} \tag{26}$$

$$\begin{gathered} \text{If} \sum_{r \in \mathcal{R} \setminus e} (x^r_{iwk}\, x^{r+1}_{wsk}) = 1 \text{ then } B_{sk} \geq B_{iwk} + c_{ws} \\ \forall w \in \mathcal{W}, s \in \mathcal{S}, i \in \{\mathcal{C}^P \cup \mathcal{C}^D \cup \mathcal{S} \setminus s\}, k \in \mathcal{K} \end{gathered} \tag{27}$$

$$\begin{gathered} z_{sk} = 1 \text{ if and only if } \sum_{p \in \mathcal{C}^P} \left( \sum_{r \in \mathcal{R}} x^r_{psk} + \sum_{r \in \mathcal{R} \setminus e} \sum_{w \in \mathcal{W}} x^r_{pwk}\, x^{r+1}_{wsk} \right) = 1 \\ \forall s \in \mathcal{S}, k \in \mathcal{K} \end{gathered} \tag{28}$$

$$\begin{aligned} &(t(s) - \eta) \sum_{r \in \mathcal{R}} \left( x^r_{gsk} + \sum_{w \in \mathcal{W}} x^r_{wsk} + \sum_{(s',s) \in A^{S-}_s} x^r_{s'sk} + \sum_{(p,s) \in A^{PS}_s} x^r_{psk} + \sum_{(d,s) \in A^{DS}_s} x^r_{dsk} \right) \leq B_{sk} \\ &\leq t(s) \sum_{r \in \mathcal{R}} \left( x^r_{sgk} + \sum_{(s,d) \in A^{SD}_s} x^r_{sdk} + \sum_{w \in \mathcal{W}} x^r_{swk} + \sum_{(s,s') \in A^{S+}_s} x^r_{ss'k} + \sum_{(s,p) \in A^{SP}_s} x^r_{spk} \right) \\ &\quad \forall s \in \mathcal{S}, k \in \mathcal{K} \end{aligned} \tag{29}$$

$$\begin{aligned} &e_p \sum_{r \in \mathcal{R}} \left( \sum_{(p,p') \in A^{P+}_p} x^r_{pp'k} + \sum_{w \in \mathcal{W}} x^r_{pwk} + \sum_{s \in \mathcal{S}_p} x^r_{psk} \right) \leq B_{pk} \\ &\leq l_p \sum_{r \in \mathcal{R}} \left( \sum_{(s,p) \in A^{SP}_p} x^r_{spk} + \sum_{(p',p) \in A^{P-}_p} x^r_{p'pk} + \sum_{(d,p) \in A^{DP}_p} x^r_{dpk} + x^r_{gpk} \right) \\ &\quad \forall p \in \mathcal{C}^P, k \in \mathcal{K} \end{aligned} \tag{30}$$

$$e_d \sum_{r\in\mathcal{R}} \left( \sum_{(d,d')\in A_d^{D+}} x^r_{dd'k} + \sum_{(d,p)\in A_d^{DP}} x^r_{dpk} + \sum_{(d,s')\in A_d^{DS}} x^r_{ds'k} + \sum_{w\in\mathcal{W}} x^r_{dwk} + x^r_{dgk} \right)$$
$$\leq B_{dk} \leq l_d \sum_{r\in\mathcal{R}} \left( x^r_{sdk} + \sum_{(d',d)\in A_d^{D-}} x^r_{d'dk} \right) \quad \forall s\in\mathcal{S}, d\in\mathcal{C}_s^D, k\in\mathcal{K} \tag{31}$$

$$0 \leq L_{sk} \leq Q \quad \forall s\in\mathcal{S}, k\in\mathcal{K} \tag{32}$$

$$x^r_{ijk} \in \{0,1\} \quad \forall (i,j)\in\mathcal{A}, r\in\mathcal{R}, k\in\mathcal{K} \tag{33}$$

$$y_{ps} \in \{0,1\} \quad \forall p\in\mathcal{C}^P, s\in\mathcal{S} \tag{34}$$

$$z_{sk} \in \{0,1\} \quad \forall s\in\mathcal{S}, k\in\mathcal{K} \tag{35}$$

The objective function (1) minimizes the total transportation cost, including the fixed costs incurred for using vehicles. Constraints (2) ensure that a vehicle leaving a supply point visits either a customer demand, a waiting station, another supply point, or the main depot $g$. The conservation of flow at supply point is completed by constraints (3). Constraints (4) and (5) represent the conservation of flow at main depot and waiting stations respectively. Constraints (6) ensure that each pickup-customer demand must be assigned to only one supply point. Constraints (7) - (9) forbid the illegal pickup legs which bring either pickup demands to the supply point to which they are not assigned or pickup demands not assigned to the same supply point.

Constraints (10) ensure that when a vehicle leaves a pickup-customer demand $p$, it goes either to another pickup-customer demand, a waiting station $w\in\mathcal{W}$, or a supply point $s\in\mathcal{W}$. These constraints together with (11) enforce the flow conservation at pickup-customer demands, and the single assignment of pickup-customer demands to legs. Similarly, constraints (12) ensure that when a vehicle leaves a delivery-customer demand $d\in\mathcal{C}_s^D$, it goes either to another delivery-customer demand of the same set $\mathcal{C}_s^D$, a pickup-customer demand $p$, a supply point $s'$, a waiting station $w$, or the main depot $g$. Constraints (12) and (13) also enforce the flow conservation at delivery-customer demands, and the single assignment of delivery-customer demands to legs.

Consistency of load variables is ensured through constraints (14) and (15), while constraints (16) enforce the restrictions on the vehicle capacity. Constraints (17) and (18) ensure that the vehicle $k$ brings load from a supply point $s$ to a delivery-customer demand $d$ of the customer zone $\mathcal{C}_s^D$ if and only if it loads freight at supply point $s$, i.e.,

$Q_s^k > 0$. Constraints (19) and (20) ensure that the vehicle $k$ goes directly from the supply point $s$ to either a pickup-customer demand $p$, any other supply point, a waiting station, or the main depot $g$ if it only unloads at $s$ and then leaves $s$ empty. Constraints (21) guarantee that the total pickup load entering each supply point equals to the total pickup demands of customers, which are assigned to the corresponding supply point.

Consistency of the time variables is ensured through constraints (22) - (27). Note that when a waiting station $w$ is reached in the $r$th position of the work assignment of vehicle $k$, the outgoing arc $(w, s)$ should be in the $(r+1)$th position of the same work assignment. Constraints (27) can be linearized by introducing new variables $v_{iwsk}^r \in \{0,1\}$ such that $v_{iwsk}^r = x_{iwk}^r x_{wsk}^{r+1}$ $\forall w \in \mathcal{W}, s \in \mathcal{S}, i \in \{\mathcal{C}^P \cup \mathcal{C}^D \cup \mathcal{S} \setminus s\}, k \in \mathcal{K}$. Constraints (27) can be made explicit by means of the following linear constraints:

$$x_{iwk}^r \geq v_{iwsk}^r \quad \forall r \in \mathcal{R}, w \in \mathcal{W}, s \in \mathcal{S}, i \in \{\mathcal{C}^P \cup \mathcal{C}^D \cup \mathcal{S} \setminus s\}, k \in \mathcal{K} \tag{36}$$

$$x_{wsk}^{r+1} \geq v_{iwsk}^r \quad \forall r \in \mathcal{R}, w \in \mathcal{W}, s \in \mathcal{S}, i \in \{\mathcal{C}^P \cup \mathcal{C}^D \cup \mathcal{S} \setminus s\}, k \in \mathcal{K} \tag{37}$$

$$x_{iwk}^r + x_{wsk}^{r+1} \leq 1 + v_{iwsk}^r \quad \forall r \in \mathcal{R}, w \in \mathcal{W}, s \in \mathcal{S}, i \in \{\mathcal{C}^P \cup \mathcal{C}^D \cup \mathcal{S} \setminus s\}, k \in \mathcal{K} \tag{38}$$

$$B_{sk} \geq B_{iwk} + c_{ws} - M(1 - \sum_{r \in \mathcal{R}} v_{iwsk}^r) \quad \forall w \in \mathcal{W}, s \in \mathcal{S}, i \in \{\mathcal{C}^P \cup \mathcal{C}^D \cup \mathcal{S} \setminus s\}, k \in \mathcal{K} \tag{39}$$

Constraints (28) ensure that the vehicle $k$ unloads at a supply point $s$ if and only if it brings pick-up demands to $s$. Because each pickup-customer demand $p$ is serviced only once, these constraints can be linearized and rewritten as follows:

$$z_{sk} = \sum_{p \in \mathcal{C}^P} \left( \sum_{r \in \mathcal{R}} x_{psk}^r + \sum_{r \in \mathcal{R}} v_{pwsk}^r \right) \quad \forall s \in \mathcal{S}, k \in \mathcal{K} \tag{40}$$

The respect of time windows at supply points and customer demands is enforced through constraints (29) - (31). Constraints (32) are bounding constraints for variables $L_s^k$. Finally, constraints (33) - (35) define the decision variables.

## 5 Solution method

In this section, we present the extension of our proposed tabu search for the TMZT-VRPTW in Nguyen et al. (2013) so that it can tackle the MZT-PDTWS. The general structure and search space are presented in Section 5.1 and 5.2, respectively. Section 5.3 describes the initial solution construction. All components of TS are then given: the neighborhood structures (Section 5.4), the neighborhood-selection *Control* procedure (Section 5.5), the tabu status mechanism (Section 5.6), the diversification mechanism (Section 5.7), and *Post-optimization* procedure (Section 5.8).

### 5.1 General structure

The structure of TS is presented in Algorithm 1. First, an initial feasible solution $z$ is generated using a greedy method seeking to fully utilize vehicles and minimize the total cost. At each iteration of the TS method, one neighborhood is selected probabilistically based on the current value of $\overline{r}$, then the selected neighborhood is explored, and the best move is chosen (lines 7-8). This move must not be tabu, unless it improves the current best TS solution $z_{best}$ (aspiration criterion). The algorithm adds the new solution to an elite set $\mathcal{E}$ if it improves on $z_{best}$. It also remembers the value of the parameter $\overline{r}$ when the new best solution was found (lines 9-13), and finally updates the elite set $\mathcal{E}$ by removing a solution based on its value and the difference between solutions (Section 5.7).

Initially, the search freely explores the solution space by assigning each neighborhood with the same probability of being selected. Whenever the best TS solution $z_{best}$ is not improved for $IT_{cNS}$ TS iterations (line 15), the *Control* procedure is called to reduce the probability of selecting leg neighborhoods (line 25). Consequently, routing neighborhoods are selected proportionally more often, which gives customer moves more opportunity to fully optimize routes. The search is re-initialized from the current best TS solution $z_{best}$ after the execution of the *Control* procedure (line 26). Moreover, after $C_{cNS}$ consecutive executions of this procedure without improvement of the current best TS solution $z_{best}$, a solution $z$ is selected randomly and removed from the elite set (line 20), and a *Diversification* mechanism is applied to perturb $z$ (line 21). The value of $\overline{r}$ is reset to the value it had when the corresponding elite solution was found, and all tabu lists are reset to the empty state (line 22). The search then proceeds from the perturbed solution $z$. The search is stopped when the elite set $\mathcal{E}$ is empty. Finally, a post-optimization procedure is performed to potentially improve the current best solution $z_{best}$ (line 30).

**Algorithm 1** Tabu search

```
1: Generate an initial feasible solution $z$
2: $z_{best} \leftarrow z$
3: Elite set $\mathcal{E} \leftarrow \emptyset$
4: Probability of selecting routing neighborhood with respect to leg neighborhood $\overline{r} \leftarrow 1$
5: STOP $\leftarrow 0$
6: repeat
7:    A neighborhood is selected based on the value of $\overline{r}$
8:    Find the best solution $z'$ in the selected neighborhood of $z$
9:    if $z'$ is better than $z_{best}$ then
10:       $z_{best} \leftarrow z'$
11:       $\overline{r}_{best} \leftarrow \overline{r}$
12:       Add $(z_{best}, \overline{r}_{best})$ to the elite set $\mathcal{E}$; update $\mathcal{E}$
13:   end if
14:   $z \leftarrow z'$
15:   if $z_{best}$ not improved for $IT_{cNS}$ iterations then
16:       if $z_{best}$ not improved after $C_{cNS}$ consecutive executions of Control procedure then
17:           if $\mathcal{E} = \emptyset$ then
18:               STOP $\leftarrow 1$
19:           else
20:               Select randomly $(z, \overline{r}_z)$ (and remove it) from the elite set $\mathcal{E}$
21:               Diversify the current solution $z$
22:               Set $\overline{r} \leftarrow \overline{r}_z$ and reset tabu lists
23:           end if
24:       else
25:           Apply Control procedure to update the value of $\overline{r}$
26:           $z \leftarrow z_{best}$
27:       end if
28:   end if
29: until STOP
30: $z_{best} \leftarrow Post\text{-}optimization(z_{best})$
31: return $z_{best}$
```

## 5.2 Search space

We allow infeasible solutions in our algorithm. Infeasible solutions are penalized in proportion to the violations of the constraints on vehicle capacity, the time windows of customer demands and supply points. More precisely, for a solution $z$, let $c(z)$ denote the total traveling cost, and let $q(z)$, $w_c(z)$ and $w_s(z)$ denote the total violation of vehicle load, customer demands time windows, supply points time windows, respectively. The total vehicle-load violation is computed on a leg basis with respect to the value $Q$, whereas the total violation of time windows of customer demands is equal to $\sum_{i \in z} max\{(a_i - l_i), 0\}$, and the total violation of time windows of supply points is equal to $\sum_{s \in z} max\{(t(s) - \eta - a_s), (a_s - t(s)), 0\}$, where $a_i$ and $a_s$ are the arrival time at customer demand $i$ and supply point $s$, respectively.

Solutions are then evaluated according to the weighted fitness function $f(z) = c(z) + \alpha^Q q(z) + \alpha^C w_c(z) + \alpha^S w_s(z) + F * m$, where $\alpha^Q$, $\alpha^C$, $\alpha^S$ are penalty parameters adjusted dynamically during the search. The updating scheme is based on the idea of Cordeau et al. (2001). At each iteration, the value of $\alpha^Q$, $\alpha^C$ and $\alpha^S$ are modified by a factor $1 + \beta > 1$. If the current solution is feasible with respect to load constraints, the value of $\alpha^Q$ is divided by $1 + \beta$; otherwise it is multiplied by $1 + \beta$. The same rule applies to $\alpha^C$ and $\alpha^S$ with respect to time window constraints of customers and supply points, respectively. In our algorithm, we set $\alpha^Q = \alpha^C = \alpha^S = 1$ and $\beta = 0.3$.

## 5.3 Initial solution

We sort the supply points and index them in increasing order of their opening times. Thus, if $t(s_1) \leq t(s_2)$, then $s_1 < s_2$ and vice versa. We then construct an initial solution by assigning each pickup-customer demand to one supply point, and building each feasible work assignment sequentially.

There are several ways to assign pickup-customer demands to supply points. A simple way is to assign each pickup-customer demand to its closest supply point. Another way is that each supply point $s$ services a predefined number of its pickup-customer demands closest to it. However, these strategies fail to take the significant variation of delivery loads at each supply point into account. The imbalance of pickup and delivery demands might happen at some supply points, which reduces the possibility of 'unload and load' operation at a supply point, and thus increases the number of empty movements.

In order to overcome this issue, we first estimate what the maximum total pickup demands at each supply point should be. As delivery-customer demands are already assigned to supply points, we calculate the total delivery demands assigned to each supply point. Let $K_s$ denote this number for each supply point $s \in S$. We then use $K_s$

as the maximum capacity of collected freight which vehicles can unload at supply point $s$ in hope of balancing the unloading and loading operations at each supply point. Each pickup-customer demand $p$ is then assigned to the nearest supply point in $\mathcal{S}_p$. When the assignment violates the maximum capacity of the nearest supply point in $\mathcal{S}_p$, it is randomly allocated to the supply point in $S_p$ whose residual capacity is large enough for the assignment. Pickup-customer demands are handled in random order. By considering both the distance from pickup-customer demands to supply points and the capacity of delivery demands at each supply point when allocating pickup-customer demands, we aim to generate a solution which satisfy the following two conditions. The first condition is a small total traveling cost. The other condition is to avoid the imbalance of the number of generated pickup legs and delivery legs at each supply point, so vehicles can do more 'unload and load' activities, which then helps to reduce empty movements.

Once the assignment of pickup-customer demands to supply points is completed, each work assignment of the initial solution is built sequentially. Each work-assignment construction consists of two phases: the first phase determines the first supply point for the current work assignment; the second phase creates sequentially each leg using a greedy algorithm.

In the first phase, the supply point $s$ with earliest opening time and unserviced customer demands is assigned as the initial supply point of the first leg of the current work assignment. During the second phase, one or a sequence of first legs between supply point $s$ and either other supply point $s'$ or the depot $g$ is created using a greedy algorithm. If the first created leg(s) ends at a supply point $s'$, we continue applying the greedy algorithm to build next leg(s) in which $s'$ is now used as the initial supply point. Otherwise, if it ends at the main depot, it means the current work assignment cannot be used anymore, and we return to the first phase to build another work assignment. This process is repeated until all customer demands are serviced (assigned to a vehicle route).

The greedy algorithm is implemented as follows: for a given initial supply point $s$ assigned to the leg, it finds a set of supply points $S' = \{s' \in \mathcal{S} | s'$ with unserviced customer demands and $t(s') > t(s)\}$. If $S' \neq \varnothing$, for each pair $(s, s')$, all unrouted customer demands of $s$ and unrouted pickup-customer demands of $s'$ are candidates for insertion to the vehicle according to a priority order which considers unrouted pickup-customer demands of $s$ first, then unrouted delivery-customer demands of $s$, and unrouted pickup-customer demands of $s'$ as the latest. Each unserviced customer demand is assigned to the vehicle sequentially by applying the heuristic I1 of Solomon (1987) until the vehicle is full.

Figure 3 illustrates different possibilities when routing customer demands between two supply points $s$ and $s'$. If there exists unrouted pickup-customer demands of $s$, the greedy algorithm assigns them to the current vehicle first, for instance, generating a pickup leg $\{p_2, p_5, p_3, s\}$. Between supply point $s$ and $s'$, the algorithm then may generate

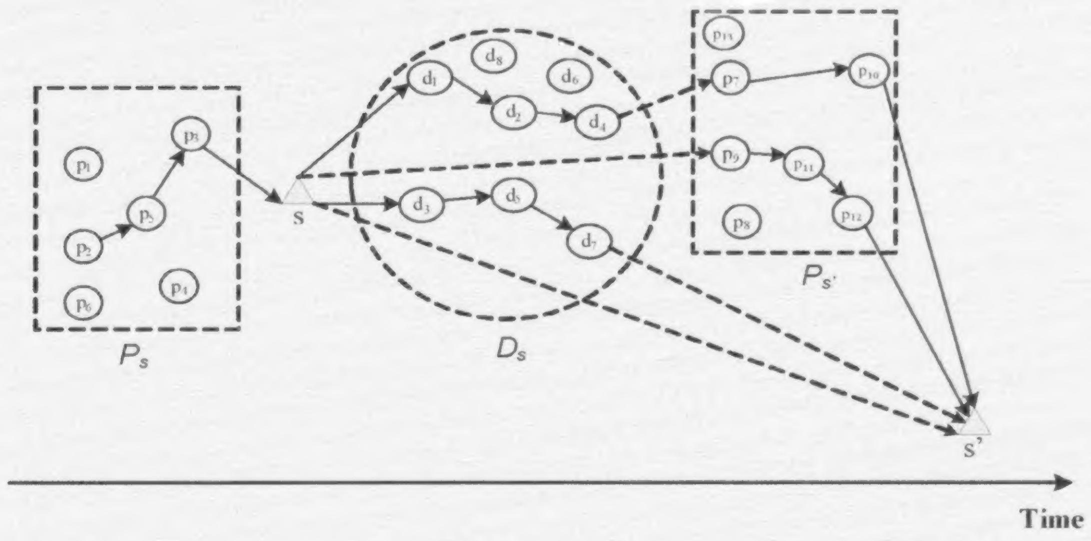


Figure 3: A generation of a sequence of legs between two supply points.

either a leg or a sequence of legs:

- (1) A sequence of two legs: a delivery leg $\{s, d_1, d_2, d_4\}$ and a pickup leg $\{p_7, p_{10}, s'\}$
- (2) A pickup leg: $\{p_9, p_{11}, p_{12}, s'\}$
- (3) A delivery leg: $\{s, d_3, d_5, d_7\}$
- (4) An empty leg connecting $s$ and $s'$

Which one is generated depends on the departure time at supply point $s$, time windows and distance between unserviced delivery- and pickup-customer demands of supply point $s$ and $s'$, respectively.

Among feasible legs generated between all pairs of $s$ and $s'$, the one with the smallest average cost per unit demand is selected and assigned to the current work assignment. The average cost per unit demand is expressed as the ratio of the total traveling time over the total demand carried by the vehicle between $s$ and $s'$. We set total demand to one for empty legs. In the case there are no feasible legs or $S' = \varnothing$, the greedy algorithm builds the last leg $(s, g)$ by applying the heuristic I1 of Solomon (1987).

## 5.4 Neighborhoods

The MZT-PDTWS has the same problem structure as the TMZT-VRPTW, where each work assignment consists of a sequence of legs and where each leg is made of a sequence of customer demands. Therefore, the MZT-PDTWS is also solved by applying neighborhoods at leg and customer levels. However, the MZT-PDTWS is more complex than the TMZT-VRPTW in the sense that a new type of customer demands, say pickup, is now considered together with delivery-customer demands. Furthermore, pickup-customer-demand-to-supply point assignments are not known in advance as for delivery-customer demands, but rather each pickup-customer demand has a list of available supply points that can service it. Consequently, it is asking for expanding neighborhoods so that the MZT-PDTWS can be addressed more efficiently. In this section, we describe in detail two types of neighborhoods, i.e., routing neighborhoods and leg neighborhoods, used in our tabu search.

### 5.4.1 Routing neighborhoods

In the MZT-PDTWS, each vehicle performs a sequence of legs, each leg services either pickup- or delivery-customer demands but cannot do both. As a result, routing neighborhoods work on two sets of pickup legs and delivery legs separately. These neighborhoods try to improve routing by using different intra and inter route neighborhoods commonly used in the VRP literature: Relocation, Exchange and 2-opt.

For delivery-customer demands whose serviced supply points are pre-assigned as those in the TMZT-VRPTW, moving them between supply points, i.e., causing reassignments to other supply points, is forbidden. Routing neighborhoods working on delivery-customer demands are therefore kept unchanged as in the TMZT-VRPTW. More precisely, for each move in each neighborhood, two delivery-customer demands which belong to a same supply point are considered:

- *Relocation move*: one of the two customer demands is taken from its current position and inserted after the other one.
- *Exchange move*: two customer demands are swapped.
- *2-opt move*: for two customer demands in the same leg, the edges emanating from them are removed, two edges are added, one of which connects these two customer demands, and the other connects their successor customer demands. For two customer demands in different legs, the remaining segments of these legs are swapped preserving the order of customer demands.

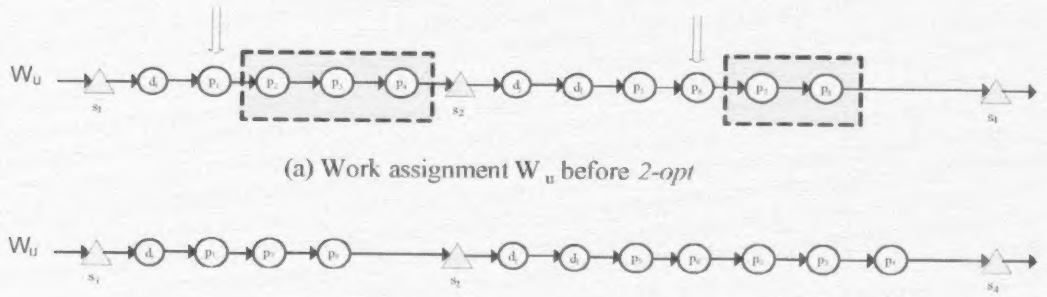
For pickup-customer demands whose lists of available serviced supply points are only given, it is required the assignment of each pickup-customer demand to a supply point selected from the given list before routing it. Consequently, we extend routing neighborhoods working on pickup-customer demands so that they could address the routing but also the supply-point assignment. It eventually helps to improve the routing through pickup-customer-demand-to-supply-point assignment and vice-versa. Since pickup-customer demands can be reassigned to other supply points, we do not restrict routing neighborhoods to work on each zone of pickup-customer demands separately as those for delivery-customer demands. However, when a move reassigns a pickup-customer demand $p$, assuming from $s_i$ to $s_j$, it is constrained to reassign the demand to a supply point that belongs to the list of available serviced supply points for $p$, i.e., $s_j \in \mathcal{S}_p$. Three types of routing neighborhoods are thus considered for all pairs of pickup-customer demands satisfying the above condition for reassignment:

- *Relocation move*: one pickup-customer demand is shifted from its current position to another position, in the same leg or in a different leg which may be assigned to the same supply point or not, provided the condition for supply-point reassignment is respected.
- *Exchange move*: two pickup-customer demands are exchanged. They may belong to the same leg or, if the condition for supply-point reassignment allow them, to two distinct legs sharing one common supply point or not.
- *2-opt move*: for two pickup-customer demands in the same leg, the edges emanating from them are removed, two edges are added, one of which connects these two pickup-customer demands, and the other connects their successor pickup-customer demands. For two pickup-customer demands in different legs sharing one common supply point, thus in different vehicles, the remaining segments of these legs are swapped preserving the order of customer demands. Finally, for two pickup-customer demands in different legs sharing two distinct supply points, i.e., in a same vehicle or different vehicles, the remaining segments of these legs are swapped preserving the order of customer demands, when the condition for supply-point reassignment is respected.

Let us take a simple example to illustrate the condition for supply-point reassignment. Consider Table 2 where the lists $\mathcal{S}_p$ for pickup-customer demands $p \in \mathcal{P}$ are given for the work assignment $W_u$ shown in Figure 4a. Consider two pickup-customer demands $p_1$ and $p_6$ in $W_u$ which belong to different supply points, $s_2$ and $s_4$, respectively. The 2-opt move of $p_1$ and $p_6$ applied on $W_u$ requires the supply-point reassignments of $\{p_2, p_3, p_4\}$ to $s_4$ and of $\{p_7, p_8\}$ to $s_2$. Customer demands $p_7$ and $p_8$ can be reassigned to supply point $s_2$ as $s_2 \in \mathcal{S}_{p_7} \cap \mathcal{S}_{p_8}$. Similarity, $p_2, p_3, p_4$ can be reassigned to $s_4$ as $s_4 \in \mathcal{S}_{p_2} \cap \mathcal{S}_{p_3} \cap \mathcal{S}_{p_4}$. The condition for supply-point reassignment is satisfied, therefore this 2-opt move is accepted. Figure 4b illustrates $W_u$ after the move.

Table 2: The lists of available serviced supply points $\mathcal{S}_p$

| Pickup-customer demand $p$ | List of available serviced supply points $\mathcal{S}_p$ |
|---|---|
| $p_1$ | $\{s_2, s_4\}$ |
| $p_2$ | $\{s_2, s_3, s_4\}$ |
| $p_3$ | $\{s_1, s_2, s_4\}$ |
| $p_4$ | $\{s_2, s_4\}$ |
| $p_5$ | $\{s_4\}$ |
| $p_6$ | $\{s_4, s_5\}$ |
| $p_7$ | $\{s_2, s_4\}$ |
| $p_8$ | $\{s_1, s_2, s_4\}$ |



(a) Work assignment W $_u$ before *2-opt*

(b) Work assignment W $_u$ after *2-opt*

Figure 4: An example of 2-opt routing neighborhood on pickup-customer demands

On the other hand, the 2-opt move of $p_1$ and $p_5$ applied on $W_u$ requires the supply-point reassignments of $\{p_2, p_3, p_4\}$ to $s_4$ and of $\{p_6, p_7, p_8\}$ to $s_2$. However, $s_2 \notin \mathcal{S}_{p_6}$, so $p_6$ can not be reassigned to supply point $s_2$. Due to the infeasibility of the supply-point reassignment, this 2-opt move is not accepted.

### 5.4.2 Leg neighborhoods

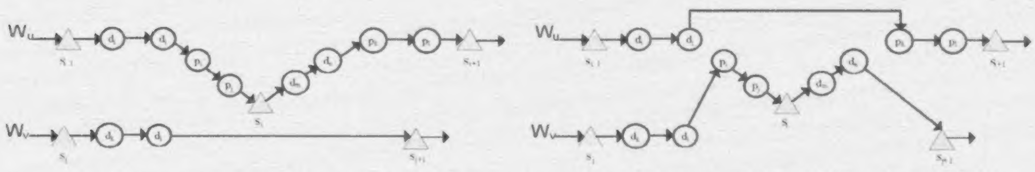
In the MZT-PDTWS, each leg is assigned to the supply point where the vehicle returns the collected freight and/or loads new freight. Let $W_u$ be the work assignment performed by vehicle $u$. Let $s_{i-1}$ and $s_{i+1}$ denote the predecessor and successor supply points, respectively, of $s_i$ within a work assignment. When moving a supply point, all customer demands serviced by it in the vehicle are also moved. Leg neighborhoods focus on repositioning legs within the time restrictions. They are described in the following:

**5.4.2.1 Relocate supply point** This neighborhood removes a supply point together with customer demands serviced by it in a work assignment and inserts them into another work assignment. Consider two work assignments $W_u$ and $W_v$. For each supply point $s_i \in W_u$:

- if $s_i \notin W_v$: for each two successive supply points $s_j$, $s_{j+1} \in W_v$, such that $s_j < s_i < s_{j+1}$, then move $s_i$ from work assignment $W_u$ to $W_v$ locating it between $s_j$ and $s_{j+1}$ (see Figure 5 for example). As we promote the 'unload and load' operation at a supply point to reduce empty movements, the reassignment of pickup-customer demands to other supply points may be required. More precisely, whenever a pickup (or a single-delivery) leg assigned to $s_i$ is relocated between $s_j$ and $s_{j+1}$, and the leg assigned to $s_{j+1}$ is a single-delivery leg (or the leg assigned to $s_j$ is pickup leg), the reassignment of all pickup-customer demands in the leg of $s_i$ (or $s_j$) to supply point $s_{j+1}$ (or $s_i$) is checked. If the reassignment is feasible, it is executed, customer demands in the leg of $s_i$ is then relocated between $s_j$ and $s_{j+1}$ to create a new 'unload and load' operation at supply point $s_{j+1}$ (or $s_i$) on the work assignment $W_v$. Otherwise, the leg assigned to $s_i$ is just simply relocated between $s_j$ and $s_{j+1}$. Figure 6 illustrates the relocation of a pickup leg assigned to $s_i$ on $W_u$ between $s_j$ and $s_{j+1}$ on $W_v$. As there are two pickup-customer demands $p_i$ and $p_j$ in this leg, and the leg assigned to $s_{j+1}$ of $W_v$ is a single delivery leg, we check whether we can reassign $p_i$ and $p_j$ to supply point $s_{j+1}$ so that vehicle $v$ can do 'unload and load' at $s_{j+1}$. If $s_{j+1} \in \mathcal{S}_{p_i} \cap \mathcal{S}_{p_j}$ then the reassignment is applied, and the movement is executed as shown in Figure 6b. Otherwise, see Figure 6c.
- if $s_i \in W_v$:
  - Case 1: if vehicle $u$ only unloads at $s_i$: this is a relocate move of the pickup leg $r_i$ assigned to $s_i$ in vehicle $u$. Three cases of vehicle's operation at supply point $s_i$ in vehicle $v$ are considered:
    - Case 1.1: if vehicle $v$ only unloads at $s_i$: denote $r_j$ the pickup leg assigned to $s_i$ in $W_v$, then move $s_i$ from work assignment $W_u$ to $W_v$ by concatenating two pickup legs $r_i$ and $r_j$. Appending $r_i$ to $r_j$ and $r_j$ to $r_i$ are both considered (see Figure 7).
    - Case 1.2: if vehicle $v$ only loads at $s_i$: denote $r_j$ the single-delivery leg assigned to $s_i$ in $W_v$, then move $s_i$ from work assignment $W_u$ to $W_v$ by locating pickup leg $r_i$ right before single-delivery leg $r_j$ so that vehicle $v$ does 'unload and load' operation at $s_i$ (see Figure 8).
    - Case 1.3: if vehicle $v$ both 'unloads and loads' at $s_i$: denote $r_j$ the pickup leg and $r'_j$ the coordinate leg assigned to $s_i$ in $W_v$, then move $s_i$ from work assignment $W_u$ to $W_v$ by concatenating two pickup legs $r_i$ and $r_j$. Both cases of appending $r_i$ to $r_j$ and $r_j$ to $r_i$ are also considered as in the case 1.1.

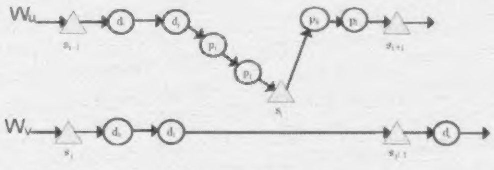
– Case 2: if vehicle $u$ only loads at $s_i$: this is a relocate move of the single-delivery leg $r_i$ assigned to $s_i$ of vehicle $u$. Three cases of vehicle's operation at supply point $s_i$ in vehicle $v$ are considered as in the Case 1. Similarly, if vehicle $v$ loads at $s_i$, i.e., there exists delivery leg $r_j$ assigned to $s_i$ in vehicle $v$, the concatenation of delivery leg $r_i$ and delivery leg $r_j$ is also examined in two cases: one appending $r_i$ to $r_j$ and the other appending $r_j$ to $r_i$.

– Case 3: if vehicle $u$ both 'unloads and loads' at $s_i$: this is a relocate move of both the pickup leg $r_i$ and the coordinate delivery leg $r'_i$ assigned to the same supply point $s_i$ in vehicle $u$. Three cases of vehicle's operation at supply point $s_i$ in vehicle $v$ are considered as in previous cases. All possibilities of concatenation delivery (pickup) legs assigned to the same supply point $s_i$ in both vehicle $u$ and $v$ are also examined.

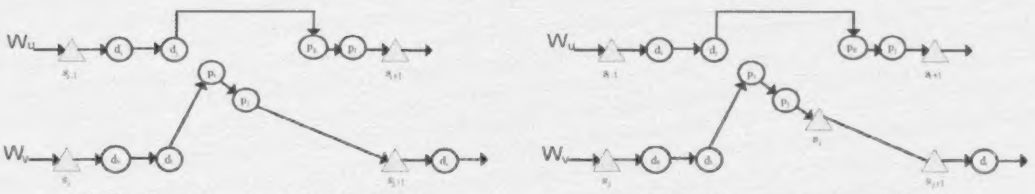


(a) Work assignments before *Relocate*    (b) Work assignments after *Relocate*

Figure 5: Relocate a supply point: relocate both pickup leg and coordinate-delivery leg assigned to a same supply point



(a) Work assignments before *Relocate*

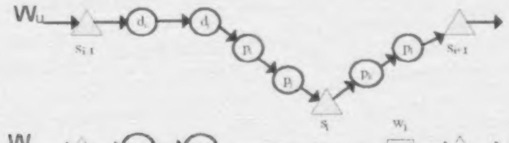


(b) Work assignments after *Relocate*
Case: supply-point reassignment is applied

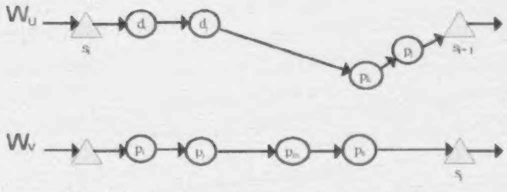
(c) Work assignments after *Relocate*
Case: supply-point reassignment is not applied

Figure 6: Relocate a supply point: relocate a pickup leg

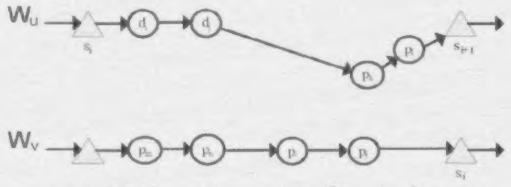
**5.4.2.2 Exchange supply point** This neighborhood exchanges legs between work assignments. Consider two work assignments $W_u$ and $W_v$. For supply points $s_i \in W_u$ and $s_j \in W_v$:



(a) Work assignments before *Relocate*

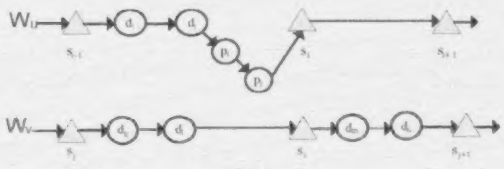


(b) Work assignments after *Relocate*
Case: append $(r_j,s_j)$ to $(r_i,s_i)$

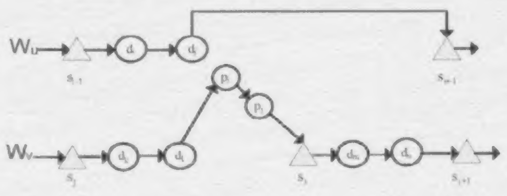


(c) Work assignments after *Relocate*
Case: append $(r_i,s_i)$ to $(r_j,s_j)$

Figure 7: Relocate a supply point: concatenation of two pickup legs



(a) Work assignments before *Relocate*



(b) Work assignments after *Relocate*

Figure 8: Relocate a supply point: creation of an 'unload and load' operation

- if $s_{i-1} < s_j < s_{i+1}$:
  - if $s_{j-1} < s_i < s_{j+1}$: simply swap $s_i$ and $s_j$ together with customer demands serviced by them (both pickup- and delivery-customer demands if any);
  - if $s_{j-1} = s_i < s_{j+1}$: first swap $s_i$ and $s_j$ together with customer demands serviced by them; next if there were pickup-customer demands assigned to $s_i$ in both vehicle $u$ and vehicle $v$, then in vehicle $v$ we concatenate pickup-customer demands from both vehicles as described in the case 1.1; and also concatenate delivery-customer demands in both vehicles if applicable;
  - if $s_{j-1} < s_i <= s_{j+1}$: same as previous case.
- otherwise if $s_{i-1} = s_j$ or if $s_j = s_{i+1}$: as supply points $s_i$ and $s_j$ play a symmetric role in this exchange leg move, these two cases are considered the same as the previous one.

The reassignment of pickup-customer demands to new supply points is considered the same as those in the relocate supply point neighborhood whenever it could create 'unload and load' operation. And only the feasible reassignments are accepted.

Moving legs or customer demands could change the traveling cost and the number of vehicles, as well as the level of constraint violations of load, time windows of customer demands, and time windows of supply points. As a result, the move value is defined as a sum of five terms $\Delta f = \Delta c + F * \Delta m + \Delta q + \Delta w_c + \Delta w_s$. The five components of the summation are the difference in traveling cost, the fixed cost of using vehicles, and the difference in violation of load, time windows at customer demands and supply points between the value of the neighboring solution and the value of the current solution.

## 5.5 Neighborhood selection strategy

The algorithm explores one neighborhood at each iteration. The neighborhood to explore is randomly selected among the five previously defined neighborhoods. As in Nguyen et al. (2013), the probability for the selection of neighborhoods is controlled by a neighborhood-selection parameter $\overline{r}$. At the beginning of the search, both leg and routing neighborhoods are given the same probability of been selected, which allows the TS algorithm to freely explore the solution space. Given that the number of supply points is much smaller than the number of customer demands in most MZT-PDTWS instances, the algorithm should perform more customer than leg moves to ensure adequate optimization of routes. Consequently, after the initial phase, the probability of selecting leg neighborhoods becomes lower than the probability of selecting routing neighborhoods. We assign to a routing neighborhood the probability $\overline{r}/(2+6\overline{r})$ of been selected, and to a leg neighborhood the probability $1/(2+6\overline{r})$ of been selected. The equal initial probabilities are then obtained by setting $\overline{r} = 1$. The *Control* procedure in our algorithm varies the value of $\overline{r}$ during execution to monotonically reduce (increase) the probability of selecting leg (routing) neighborhoods after each $IT_{cNS}$ iterations without improvement of the best solution. A linear scheme $\overline{r}_{k+1} = \overline{r}_k + \Delta\overline{r}$ is used, where $\Delta\overline{r}$ is a user defined parameter, $\overline{r}_{k+1}$ and $\overline{r}_k$ are values of $\overline{r}$ at iteration $k+1$ and $k$, respectively.

## 5.6 Tabu lists and tabu duration

We keep a separate tabu list for each type of move. Elements of a solution generated by a move are given a tabu status as follows:

- Leg moves:

  - Relocation move: the position of supply point $s_i$ just inserted into work assignment $W_v$ cannot be changed by another relocate supply point move while it is tabu.
  - Exchange move: supply points $s_i$ and $s_j$ just swapped cannot be swapped again while they are tabu.

- Customer moves:
  - Relocation move: the position of customer demand $i$ just inserted after customer demand $j$, cannot be changed by the same type of move while it is tabu.
  - Exchange move: customer demands $i$ and $j$ just swapped cannot be swapped again while they are tabu.
  - 2-opt move: a 2-opt move applied to customer demands $i$ and $j$ cannot be applied again to the same customer demands while tabu.

A tabu status is assigned to each tabu list element for $\theta$ iterations, where $\theta$ is randomly selected from a uniform interval. Any move declared tabu cannot be performed unless it yields a solution which improves the current best solution. Generally, the tabu status of a move stays so for a number of iterations proportional to the number of possible moves. Consequently, we use different intervals of the tabu list size for leg and routing moves. Since there are $O(m' * |\mathcal{S}|)$ possible leg moves, we set the interval of tabu list size for leg moves to $[m'*|\mathcal{S}|/a_1, m'*|\mathcal{S}|/a_2]$, where $m'$ is the number of vehicles used in the initial solution, and $a_1$ and $a_2$ are user-defined parameters.

In MZT-PDTWS, each delivery-customer demand is serviced by a fixed supply point which is known in advance. Therefore, the number of iterations during which a customer demand move within the delivery customer zone of a supply point $s$ remains tabu is only counted each time the algorithm deals with customer demands in that zone. The interval of tabu list size for delivery-customer demand moves for each supply point $s$ with $|\mathcal{C}_s^D|$ associated customers is therefore calculated as $[a_3 log_{10}(|\mathcal{C}_s^D|), a_4 log_{10}(|\mathcal{C}_s^D|)]$, where $a_3$ and $a_4$ are user defined parameters. As pickup-customer demands are not fixed to any supply points yet, the interval of tabu list size for pickup-customer demand moves is $[a_5 log_{10}(|\mathcal{C}^P|), a_6 log_{10}(|\mathcal{C}^P|)]$, where $a_5$ and $a_6$ are user defined parameters.

## 5.7 Diversification strategy

A diversification strategy, based on an elite set and a frequency-based memory, direct the search to potentially unexplored promising regions when the search begins to stagnate. In a nutshell, diversification aims to capitalize on the best attributes obtained so far by selecting a new working solution from the elite set and perturbing it based on long-term trends.

In more details, we use the elite set as a diversified pool of high-quality solutions found during the tabu search. The elite set starts empty and is limited in size. The quality and diversity of the elite set is controlled by the insertion of new best solutions produced by the tabu search and the elimination of the existing solutions in the elite set. The elimination is based on the Hamming distance $\Delta(z_1, z_2)$ measuring not only the number of customer demand positions that differ between solutions $z_1$ and $z_2$ as in the TMZT-VRPTW, but also the differences between supply-point assignments of pickup-customer demands. More precisely, this distance is computed according to Equation (41), where $\mathbf{T}(cond)$ is a valuation function that returns 1 if the condition $cond$ is true, 0, otherwise; $N_z[i]$ is the next place (which is either a customer demand, the depot, or a supply point) visited by the vehicle after servicing customer demand $i$ in solution $z$; and $S_z[i]$ is the supply point assigned to pickup-customer demand $i$ in the solution $z$.

$$\Delta(z_1, z_2) = \sum_{i \in \mathcal{C}^P \cup \mathcal{C}^D} \mathbf{T}(N_{z_1}[i] \neq N_{z_2}[i]) + \sum_{i \in \mathcal{C}^P} \mathbf{T}(S_{z_1}[i] \neq S_{z_2}[i]) \tag{41}$$

The elimination of a solution from the elite set is considered each time a new best solution $z_{best}$ is inserted. There are two cases. If the elite set is not yet full, we delete only when there exists a solution very similar to the new $z_{best}$, i.e., we delete the solution $z$ with the smallest $\Delta(z, z_{best}) \leq 0.05(|\mathcal{C}^D| + 2|\mathcal{C}^P| + |\mathcal{S}|)$. When the elite set is full, $z_{best}$ replaces the solution $z$ that is the most similar to it, i.e., the one with the smallest $\Delta(z, z_{best})$.

The long-term frequency memory keeps a history of the arcs most frequently added to the current solution as well as the supply-point assignments of pickup-customer demands most frequently used. Let $t_{ij}$ be the number of times arc $(i, j)$ has been added to the solution during the search process. The frequency of arc $(i, j)$ is then defined as $\rho_{ij} = t_{ij}/T$, where $T$ is the total number of iterations executed so far. Similarly, let $t'_{ps}$ be the number of times pickup-customer demand $p$ has been assigned to supply point $s$ during the search. The frequency of the supply-point assignment of customer demand $p$ to $s$ is defined as $\chi_{ps} = t'_{ps}/T$.

Diversification then proceeds to perturb the search that starts from the solution taken from the elite set by removing arcs with high frequency and inserting arcs with low frequency and promoting never-seen supply-point assignments. Thus, the evaluation of neighbor solutions is biased so as to penalize the arcs most frequently added to the current solution and the supply-point assignment most frequently used.

More precisely, the corresponding two penalties $g_1(\bar{z})$ and $g_2(\bar{z})$ are added to the evaluation of the fitness $f(\bar{z})$ (Section 5.2) of a neighbor $\bar{z}$ of the current solution $z$:

$$g_1(\bar{z}) = \bar{C}\Big( \sum_{(i,j)\in A_a} \rho_{ij} + \sum_{(i',j')\in A_r} (1 - \rho_{i'j'})\Big) \tag{42}$$

$$g_2(\bar{z}) = \bar{C} \sum_{p\in C^P} \left[ \sum_{\substack{s\in\mathcal{S}_p \\ S_z(p)=S_{\bar{z}}(p)=s}} \chi_{ps} + \sum_{\substack{s\in\mathcal{S}_p \\ S_z(p)\neq s \\ S_{\bar{z}}(p)=s}} \chi_{ps} + \sum_{\substack{s\in\mathcal{S}_p \\ S_z(p)=s \\ S_{\bar{z}}(p)\neq s}} (1 - \chi_{ps}) \right] \tag{43}$$

where $\bar{C}$ is the average cost of all arcs in the problem, and $A_a$ and $A_r$ are the sets of arcs that are added to and removed from the solution $z$ in the move to $\bar{z}$, respectively.

In this way, we introduce into the solution new arcs and supply-point assignments which helps to direct the search into unexplored regions. The diversification mechanism is executed $IT_{div}$ iterations.

## 5.8 Post optimization

The best solution obtained through the tabu search is enhanced by applying a local-search *Supply-point-improvement* procedure and a *Leg-improvement* procedure sequentially. The purpose of implementing two procedures is to improve the routing and the supply-point assignment.

The *Supply-point-improvement* proceeds by assigning a new supply point to each pickup-customer demand, keeping those that actually improve the solution. Pickup-customer demands are handled in random order. Then, for each pickup-customer demand $p$ and each of its unassigned supply point $s \in \mathcal{S}_p$ (if any), $p$ is removed from its current legs (i.e., current assigned supply point) and the cheapest fitness insertion is performed to insert $p$ into each pickup leg assigned to $s$. The best feasible improvement one is executed (if any). One then proceeds to the next unassigned supply point or, if all have been tried out, to the next pickup-customer demand.

The *Leg-improvement* is then performed by applying a number of well-known local-search route improvement techniques. Two are intra-route operators, the 2-opt of Lin (1965) and the Or-opt of Or (1976). The others are inter-route operators, the $\lambda$-interchange of Osman (1993), and the CROSS-exchange of Taillard et al. (1997). For the $\lambda$-interchange, we only consider the cases where $\lambda = 1$ and $\lambda = 2$ corresponding to the (1,0), (1,1), (2,0), (2,1), and (2,2)-interchange operators. A delivery-customer demand is re-allocated only to legs with the same initial supply point. This procedure is therefore executed for each delivery customer zone separately. For pickup-customer demands which could change their supply points, the procedure is executed for all pairs of pickup-customer demands satisfying the supply-point assignment.

The procedure starts by applying in random order the five $\lambda$-interchange and CROSS-exchange inter-route operators. Each neighborhood is searched on all possible pairs of legs (in random order) and stopped on the first feasible improvement. The solution is then modified and the process is repeated until no further improvement can be found. The search is then continued by locally improving each leg of each vehicle in turn. The intra-route 2-opt and Or-opt neighborhoods are sequentially and repeatedly applied until no more improvement is found.

## 6 Experiments

Because our problem is new, no benchmark instances are available for it. We have first created MZT-PDTWS test instances from known TMZT-VRPTW benchmark problems. Next, we have studied the impact of a number of major parameters and search strategies on the performance of the proposed algorithm in order to identify the best design. We then have analyzed the impact of sharing the same fleet of vehicles and synchronization schemes on solution quality. Finally, in order to evaluate the performance of method, we made comparisons with published results of the VRPB with and without time windows.

Our tabu search algorithm is implemented in C++. Experiments were run on a 2.8 GHz Intel Xeon 4-core processor with 16GB of RAM.

### 6.1 Test data generation

We have generated new data sets for our problem based on the TMZT-VRPTW instances proposed in Crainic et al. (2009) and two given parameters, the ratio $BH = |\sum_{p \in C^P} q_p / \sum_{i \in C^P \cup C^D} q_i|$ - the total demand of backhauls over the total demand of backhauls and linehauls, and the value $M_{SP} = max_{p \in C^P} \|S_p\|$.

Based on the hypothesis that the volume of goods moving out the city is relatively lower than the volume of goods moving in, we have set the values of $BH$ at $\{0.1, 0.3, 0.5\}$. For the sake of simplification, we have also used $BH$ as the ratio of the number of backhauls over the total number of backhauls and linehauls.

We have generated six sets of 15 instances each, for a total of 90 problem instances. The six sets are called A1, A2, B1, B2, C1, and C2. Each set is further divided into three groups of 5 instances, each group is defined by one of the three different values of $BH = \{0.1, 0.3, 0.5\}$. Table 3 summarizes the parameters of all the MZT-PDTWS instances. The last column lists the names of the TMZT-VRPTW instances used to create our new MZT-PDTWS benchmark.

Each MZT-PDTWS instance is constructed from a TMZT-VRPTW to which a set of new pickup-customer demands has been added, while all delivery-customer demands, supply points, waiting stations and their attributes in the TMZT-VRPTW are kept unchanged. The added pickup-customer demands are generated based on a value of $BH$. The attributes of each pickup-customer demand $p$ are generated as follows:

- The coordinates $[X_p, Y_p]$ are uniformly distributed in the same interval used to generate coordinates of the delivery-customer demands.
- The volume of demand $q_p$ is randomly generated in the same interval as for delivery-customer demands, i.e., [5, 25], with respect to the value of $BH$.
- The service time $\delta(p)$ is set to 20 as in TMZT-VRPTW.
- The number of available supply points assigned to pickup-customer demand $p$ is selected randomly in the range [1, $M_{SP}$]. Let $x$ denote this number. Then, the list of available supply points assigned to $p$ is determined by randomly selecting $x$ supply points in the problem.
- Time window $[e_p, l_p]$: Assuming $s_1, s_2, ..., s_x$ are $x$ supply points available to service pickup-customer demand $p$ in increasing order of opening times. To ensure feasibility, the values of $e_p$ and $l_p$ are then chosen randomly in the interval [$E_p$ - 300, $E_p$] and [$L_p$ - 300, $L_p$], respectively, where $E_p = t(s_1) - \delta(p) - \lceil c_{p,s_1} \rceil$ and $L_p = t(s_x) - \delta(p) - \lceil c_{p,s_x} \rceil$.

All other attributes are kept the same as in the TMZT-VRPTW instances. The numbers of supply points (waiting stations) for the six sets are 4(4), 8(4), 16(16), 32(16), 36(36), and 72(36), respectively. Supply points, waiting stations, and customers are uniformly distributed in a square, with the X and Y coordinates in the interval [0, 100], [0, 200], and [0,300] for set of type A, B, and C, respectively. The opening times of supply points are generated randomly in the [1000, 15,400] range, while the limited allowable waiting time at supply points $\eta = 100$. The vehicle-loading and vehicle-unloading times at supply points are set to 30, for all supply points. The fixed cost and the capacity of each vehicle are set to 500 and 100, respectively, for all instance sets.

## 6.2 Algorithm design and calibration

We have aimed for a general algorithmic structure avoiding instance-related parameter settings. We have therefore defined settings as functions of problem size for the main

Table 3: Summary of the benchmark test

| Problem set | Instances name | Number of supply points | Number of waiting stations | $BH$ | Number of customers | | [X,Y] customer coordinates | $M_{SP}$ | Original instances by Crainic et al. (2009) |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | Linehauls | Backhauls | | | |
| | A1-1 ... A1-5 | | | 0.1 | | 44 | | | |
| A1 | A1-6 ... A1-10 | 4 | 4 | 0.3 | 400 | 171 | [0,100] | 2 | A1-1 ... A1-5 |
| | A1-11 ... A1-15 | | | 0.5 | | 400 | | | |
| | A2-1 ... A2-5 | | | 0.1 | | 44 | | | |
| A2 | A2-6 ... A2-10 | 8 | 4 | 0.3 | 400 | 171 | [0,100] | 2 | A2-1 ... A2-5 |
| | A2-11 ... A2-15 | | | 0.5 | | 400 | | | |
| | B1-1 ... B1-5 | | | 0.1 | | 177 | | | |
| B1 | B1-6 ... B1-10 | 16 | 16 | 0.3 | 1600 | 685 | [0,200] | 3 | B1-1 ... B1-5 |
| | B1-11 ... B1-15 | | | 0.5 | | 1600 | | | |
| | B2-1 ... B2-5 | | | 0.1 | | 177 | | | |
| B2 | B2-6 ... B2-10 | 32 | 16 | 0.3 | 1600 | 685 | [0,200] | 3 | B2-1 ... B2-5 |
| | B2-11 ... B2-15 | | | 0.5 | | 1600 | | | |
| | C1-1 ... C1-5 | | | 0.1 | | 400 | | | |
| C1 | C1-6 ... C1-10 | 36 | 36 | 0.3 | 3600 | 1542 | [0,300] | 4 | C1-1 ... C1-5 |
| | C1-11 ... C1-15 | | | 0.5 | | 3600 | | | |
| | C2-1 ... C2-5 | | | 0.1 | | 400 | | | |
| C2 | C2-6 ... C2-10 | 72 | 36 | 0.3 | 3600 | 1542 | [0,300] | 4 | C2-1 ... C2-5 |
| | C2-11 ... C2-15 | | | 0.5 | | 3600 | | | |

parameters of the proposed algorithm, tabu tenure, neighborhood selection-control.

### 6.2.1 Tabu tenure calibration

The intervals for the tabu list tenures for leg, delivery, and pickup routing moves were defined in Section 5.6 as $[m'^*|\mathcal{S}|/a_1, m'^*|\mathcal{S}|/a_2]$, $[a_3 log_{10}(|\mathcal{C}^D_s|), a_4 log_{10}(|\mathcal{C}^D_s|)]$, and $[a_5 log_{10}(|\mathcal{C}^P|), a_6 log_{10}(|\mathcal{C}^P|)]$, respectively. Using a large interval for routing moves, [10, 20], we tested different values for $a_1$ in the integer interval [7, 10] and for $a_2$ in the integer interval [4, 6]. We have observed that too large an interval is not productive as low values cannot prevent cycling, while high ones overly restrict the search path. We have therefore set $a_1$ and $a_2$ to 7 and 5, respectively.

A similar process has been used to explore different values of $a_3$, $a_4$, $a_5$, $a_6$ in the integer interval [4, 6], [7, 9], [6, 8] and [10, 12], respectively, using delivery and pickup routing tabu as defined above. We have used a larger value of tabu tenure for routing moves on pickup-customer demands as they are not restricted to one customer zone as those on delivery-customer demands. We found that the most appropriate values for $a_3$, $a_4$, $a_5$ and $a_6$ are 6, 8, 7 and 10, respectively.

### 6.2.2 Calibration of the neighborhood selection probabilities

Adjustments to the neighborhood selection probabilities depend on two parameters: $IT_{cNS}$, the number of consecutive iterations without improvement of the best solution (this number triggers the execution of the *Control* procedure that modifies probabilities), and $\Delta\overline{r}$, the adjustment factor of the neighborhood-selection parameter $\overline{r}$.

The value of $IT_{cNS}$ is defined as a function of the problem size. This value should be large enough to give each customer and supply point in each leg the possibility to be moved. Thus, $IT_{cNS} = e_1 * (m' * |\mathcal{S}| + n)$, where $m'$ is the number of vehicles used in the initial solution, $|\mathcal{S}|$ and $n$ are the numbers of supply points and customer demands, respectively, and $e_1$ is a user defined parameter. Similarly, $\Delta\overline{r}$, the amplitude of the modifications in the probabilities, is set to be proportional to the ratio of the number of customer demands with the number of supply points. Thus, $\Delta\overline{r} = e_2 \log_{10}(n/|S|)$, where $e_2$ is a user defined parameter.

Searching for a good combination of values for $e_1$ and $e_2$ concerns balancing between exploration and exploitation. On one hand, the higher the value of $IT_{cNS}$, the more chances customers and supply points are to be moved between routes, thus favoring exploration. On the other hand, a too high $IT_{cNS}$ value may waste time in useless moves. We have experimented with different values of $e_1$ in the integer interval [1,5] and $e_2$ in the integer interval [1, 7]. Three runs were performed for each instance for one million iterations. Computational results for each combination of values $(e_1, e_2)$ over all instances are summed up in Table 4, which displays the average gaps between the best solutions obtained by each combination and the best combination.

Table 4: Performance comparison between $(e_1, e_2)$ combinations

| $e_1$ | $e_2$ | | | | | | |
|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 | 1.25% | 1.04% | 0.43% | 0.34% | 0.32% | 0.28% | 0.28% |
| 2 | 1.14% | 0.98% | 0.21% | 0.23% | 0.26% | 0.31% | 0.31% |
| 3 | 1.12% | 0.73% | 0.09% | 0.06% | 0% | 0.08% | 0.17% |
| 4 | 0.97% | 0.71% | 0.14% | 0.08% | 0.04% | 0.18% | 0.21% |
| 5 | 1.05% | 0.68% | 0.12% | 0.07% | 0.05% | 0.17% | 0.28% |

Table 4 indicates that (3,5) is the most appropriate combination for $(e_1, e_2)$, giving best solutions on average. We have also observed that executing the algorithm with $\overline{r}$ greater than 60 $\log_{10}(n/|S|)$ yields an average improvement of the best solution of less than 0.1%, while requiring about 41% more time. Based on these results, we used $(e_1, e_2) = (3, 5)$ and $\overline{r}_{max} = 60 \log_{10}(n/|S|)$, the maximum value of $\overline{r}$, in the remaining experiments.

### 6.2.3 Elite set calibration, diversification

We now turn to the parameters characterizing the diversification procedure and the elite set utilization, and examine their impact on the performance of the algorithm. Four variants of the algorithm were studied corresponding to the different ways to set an elite solution as the new working solution and the inclusion, or not, of the diversification phase. The first two variants simply select an elite solution $z$ at random and re-start the algorithm from it. The *Diversification* mechanism described in Section 5.7 is applied in the last two variants to diversify from the elite solution $z$.

The initialization of the $\overline{\tau}$ parameter following the selection of $z$ is a component common to the four variants. We have studied two alternatives where $\overline{\tau}$ was set to either the full or half the value at which $z$ was found, respectively (i.e., $\overline{\tau} = \overline{\tau}_z$ or $\overline{\tau} = \overline{\tau}_z/2$). The size of the elite set is relevant for the *Diversification* mechanism only. Three values were tested, 1, 5, and 10.

Similar to previous experiments, we have used formulas dependent on the problem dimensions for $IT_{div}$ and $C_{cNS}$, which determine for how long exploration can proceed. Thus, the number of diversification phases is set to $IT_{div} = m' * |\mathcal{S}| + n$, where $m'$ is the number of vehicles used in the initial solution, and $|\mathcal{S}|$ and $n$ are the numbers of supply points and customer demands, respectively.

We have also set the number of consecutive executions of the *Control* procedure without improvement of the best solution to $C_{cNS} = min(3\log_{10}(n/|S|), (\overline{\tau}_{max} - \overline{\tau})/\Delta_{\overline{\tau}})$, which keeps the value of $C_{cNS}$ sufficiently high during the course of the algorithm, even though *Control* procedure is started with different values of $\overline{\tau}$ (remember that $\overline{\tau}_{max} = 60\log_{10}(n/|S|)$). Intuitively, in the beginning, $\overline{\tau}$ is small and $C_{cNS}$ takes the value $3\log_{10}(n/|S|)$, while when $\overline{\tau}$ becomes large enough, $C_{cNS}$ takes the value $(\overline{\tau}_{max} - \overline{\tau})/\Delta_{\overline{\tau}}$.

Table 5 displays the performance comparison between the four variants with the three different values for the elite set size. For each variant and size of the elite set, the table shows the average gaps to the cost of the best solutions obtained by it from those obtained by the case without using the elite set and diversification, together with the corresponding average computation time in minutes over 10 runs.

As expected, results indicate that guidance using elite solutions contributes significantly to improve the performance of the algorithm. Without using the elite set, the algorithm requires the lowest computation effort but produces worst solutions compared to all the variants using the elite set. Comparing the two variants corresponding to the two values at which $\overline{\tau}$ is reset, one observes that the solution quality is not very sensitive to this value, but computing effort is increasing when the value of $\overline{\tau}$ is lower ($\overline{\tau} = \overline{\tau}_z/2$).

One observes that the third and fourth variants are significantly better in terms of

Table 5: Performance comparison between diversification settings

| Elite set size | Without diversification | | | | With diversification | | | |
|---|---|---|---|---|---|---|---|---|
| | 1st variant | | 2nd variant | | 3rd variant | | 4th variant | |
| | $r = r_z$ | | $r = r_z/2$ | | $r = r_z$ | | $r = r_z/2$ | |
| | GAP (%) | Time | GAP (%) | Time | GAP (%) | Time | GAP (%) | Time |
| 0 | 0 | 50 | - | - | - | - | - | - |
| 1 | -0.37 | 66 | -0.36 | 92 | -1.02 | 88 | -1.05 | 103 |
| 5 | -0.64 | 95 | -0.69 | 117 | -1.54 | 157 | -1.48 | 194 |
| 10 | -0.78 | 121 | -0.74 | 139 | -1.55 | 223 | -1.50 | 260 |

finding high quality solutions. This indicates that the long-term memory and the diversification mechanism added to the algorithm are important features for high performance. Moreover, setting the size of the elite set to 5 achieves a better balance between solution quality and computation time, compared to a larger size of 10. Indeed, doubling the size of the elite set improves only slightly the solution quality, 0.01%, but requires 42% more time. We therefore set the size of the elite set to 5 and reset $\overline{r} = \overline{r}_z$.

## 6.3 Numerical results

Table 6 displays the results obtained by the proposed tabu search meta-heuristic over 10 runs for each group of instances. It gives the average results (*Avg 10* column), the best results (*Best 10* column), the number of vehicles (*# Vehicles* column), the percentage of times vehicles move directly to supply points without using waiting stations (*DM(%)* column), and the percentage of times vehicles do both unload and load once they arrive at supply points (*PD(%)* column). Average computation times in minutes are displayed in the *Time* column.

Experiment results show that, in total, 4874 vehicles are used in the 90 problem instances, servicing a total of 39790 legs. Hence, on average, each vehicle services 8 legs. Table 6 shows that the percentage of times vehicles do both unload and load increases proportionally to the percentage of pickup-customer demands (i.e., the value of BH). On average, 49.88% of times the vehicles do both unload and load once they arrive at supply points. This high, 49.88%, factor of 'unload and load' operations at supply points not only reduces the number of empty moves but also the traveling cost. Moreover, experiments show that the traveling cost and the number of vehicles of initial solutions are 32.45% and 20.76% greater than those of best solutions on average, respectively, illustrating the significant solution-improvement effect of the proposed algorithm.

Table 6: Performance of TS on all instances

| Problem set | BH | Avg 10 | Best 10 | #Vehicles | DM (%) | PD (%) | Time(min) |
|---|---|---|---|---|---|---|---|
|  | 0.1 | 19873.29 | 19758.67 | 21.8 | 10.45 | 21.89 | 20 |
| A1 | 0.3 | 21007.60 | 20854.25 | 22 | 27.44 | 60.93 | 34 |
|  | 0.5 | 23455.87 | 23245.62 | 22.2 | 51.29 | 87.1 | 58 |
|  | 0.1 | 16884.05 | 16756.85 | 16.4 | 14.77 | 21.52 | 12 |
| A2 | 0.3 | 18462.56 | 18295.76 | 16.4 | 31.75 | 56.75 | 19 |
|  | 0.5 | 21150.77 | 20981.06 | 17.2 | 45.28 | 88.05 | 33 |
|  | 0.1 | 66979.79 | 66763.80 | 46.8 | 19.33 | 15.01 | 66 |
| B1 | 0.3 | 75587.05 | 75398.22 | 47.8 | 31.3 | 46.73 | 139 |
|  | 0.5 | 99155.77 | 99025.96 | 54.8 | 38.31 | 80.39 | 231 |
|  | 0.1 | 59828.68 | 59717.48 | 36.4 | 19.06 | 16.53 | 42 |
| B2 | 0.3 | 72098.73 | 71945.56 | 40 | 23.64 | 46.76 | 97 |
|  | 0.5 | 94024.35 | 93838.52 | 46 | 32.63 | 78.41 | 198 |
|  | 0.1 | 153335.20 | 153106.40 | 90.4 | 17.65 | 13.84 | 172 |
| C1 | 0.3 | 200072.40 | 199848.80 | 99.4 | 21.78 | 46.01 | 310 |
|  | 0.5 | 292032.84 | 291836.60 | 119.8 | 30.91 | 82.58 | 705 |
|  | 0.1 | 141018.12 | 140803.04 | 76.2 | 18.26 | 15.65 | 112 |
| C2 | 0.3 | 195573.18 | 195206.00 | 94.4 | 24.59 | 41.92 | 213 |
|  | 0.5 | 278354.82 | 278058.20 | 106.8 | 26.45 | 77.77 | 348 |
| Average |  | 102716.39 | 102524.49 | 54.16 | 26.94 | 49.88 | 156.06 |

## 6.4 The benefits of combining linehauls and backhauls

The combining of linehaul- and backhaul-customer demands on each vehicle is expected to reduce the total number of vehicles and total traveling cost with respect to the case that services the linehauls and backhauls on separate vehicles. Table 7 compares the best solutions of both alternatives for all instances over 10 runs. In this table, LH-BH refers to the solutions with linehaul- and backhaul-customer demands on the same vehicles, while LH+BH refers to the solutions with linehaul- and backhaul-customer demands on different vehicles. The LH+BH solutions can be seen as the summation of solutions to the problem in two cases, one dealing with only linehaul customers and the other dealing with only backhaul customers. The average number of vehicles, traveling cost, and total cost of LH-BH solutions on each set of problems are given in column *#Vehicles*, *Traveling cost*, and *Total cost*, respectively. For the column 'LH+BH', each entry consists of three numbers indicating the gaps between attributes of solutions obtained by LH+BH and those obtained by LH-BH. The first number displays the gap to the average number of vehicles obtained by the LH+BH strategy from the average number of vehicles obtained by the LH-BH strategy, while the second and third numbers indicate the gaps of average traveling cost and total cost, respectively.

As expected, results indicate that assigning linehauls and backhauls to separate fleet of vehicles leads to an increase in both the average number of vehicles and traveling cost. This increase becomes significant when more backhauls are serviced, i.e., higher value of BH. In all cases, an increase in total cost is thus also observed, with an average gap of 27.63% and a maximal gap of 62.48%.

## 6.5 Synchronization at supply points

Each supply point is defined as a combination of a satellite and an availability time period in our problem. Thus the vehicles must arrive at supply points during these predefined periods to unload and/or load freight. In this section, we analyze the impact of synchronization of vehicles' operations requirement at supply points on solution quality.

In all previous experiments, the requirement for availability of vehicle at each supply point $s$ is characterized by only one time window $[e_s, l_s]$ of $s$ which is used for both unload and load operations. In order to analyze the impact of available requirements without modifying time windows at customer demands, we introduce into the model two time windows for unloading and loading respectively at each supply point, but keep the availability time periods of supply points unchanged. More precisely, we use $[e_s^u, l_s^u]$ and $[e_s^l, l_s^l]$, specifying the earliest and latest times at which the vehicle has to be available at $s$ for unloading collected demands and loading delivery demands, respectively, where $l_s^u + \varphi'(s) \leq l_s^l$, $e_s^u = e_s$ and $l_s^l = l_s$. Activities of a vehicle at $s$ are then described as

Table 7: Comparison of separate and combined linehaul and backhaul solutions in number of vehicles, traveling cost, and total cost

| Problem set | BH | LH-BH | | | LH+BH | | |
|---|---|---|---|---|---|---|---|
| | | #Vehicles | Traveling cost | Total cost | GAP (%) | | |
| A1 | 0.1 | 21.8 | 8858.67 | 19758.67 | 12.84 | 9.08 | 11.16 |
| | 0.3 | 22 | 9854.25 | 20854.25 | 45.45 | 26.25 | 36.38 |
| | 0.5 | 22.2 | 12145.62 | 23245.62 | 89.19 | 38.06 | 62.48 |
| A2 | 0.1 | 16.4 | 8556.85 | 16756.85 | 13.41 | 10.14 | 11.74 |
| | 0.3 | 16.4 | 10095.76 | 18295.76 | 35.37 | 21.91 | 27.94 |
| | 0.5 | 17.2 | 12381.06 | 20981.06 | 69.77 | 34.57 | 49.00 |
| B1 | 0.1 | 46.8 | 43363.80 | 66763.80 | 8.55 | 11.52 | 10.48 |
| | 0.3 | 47.8 | 51498.22 | 75398.22 | 35.98 | 29.50 | 31.55 |
| | 0.5 | 54.8 | 71625.96 | 99025.96 | 48.18 | 33.71 | 37.71 |
| B2 | 0.1 | 36.4 | 41517.48 | 59717.48 | 10.99 | 10.65 | 10.76 |
| | 0.3 | 40 | 51945.56 | 71945.56 | 38.00 | 28.73 | 31.31 |
| | 0.5 | 46 | 70838.52 | 93838.52 | 57.39 | 29.76 | 36.53 |
| C1 | 0.1 | 90.4 | 107906.40 | 153106.40 | 0.22 | 14.68 | 10.41 |
| | 0.3 | 99.4 | 150148.80 | 199848.80 | 24.55 | 28.87 | 27.79 |
| | 0.5 | 119.8 | 231936.60 | 291836.60 | 39.07 | 26.11 | 28.77 |
| C2 | 0.1 | 76.2 | 102703.04 | 140803.04 | 0.79 | 22.56 | 16.67 |
| | 0.3 | 94.4 | 148006.00 | 195206.00 | 10.59 | 31.48 | 26.43 |
| | 0.5 | 106.8 | 224658.20 | 278058.20 | 35.39 | 29.00 | 30.23 |
| Average | | 54.16 | 75446.71 | 102524.49 | 31.98 | 24.25 | 27.63 |

follows:

- Only unload at $s$: the vehicle arrives at $s$ with pickup demands at time $t$ within its unload time window $[e^u_s, l^u_s]$, i.e., the vehicle must not arrive at $s$ sooner than $e^u_s$ and no later than $l^u_s$; it takes $\varphi'(s)$ for unloading all demands, the vehicle thus leaves $s$ empty at time $t + \varphi'(s)$;
- Only load at $s$: the vehicle arrives at $s$ empty at time $t$ within its load time window $[e^l_s, l^l_s]$, i.e., the vehicle must not arrive at $s$ sooner than $e^l_s$ and no later than $l^l_s$; it takes $\varphi(s)$ for loading delivery demands, with a total load not exceeding $Q$; the vehicle then leaves $s$ at time $t + \varphi(s)$ to perform the delivery for serving a subset of delivery customers in $\mathcal{C}^D_s$;
- Unload and load at $s$: the vehicle arrives at $s$ with pickup demands at time $t$ within its unload time window $[e^u_s, l^u_s]$; it takes $\varphi'(s)$ for unloading all demands; in case $t + \varphi'(s) < e^l_s$, the vehicle has to wait at supply point till $e^l_s$ to start loading freight; otherwise it starts to load freight at $t + \varphi'(s)$; it takes $\varphi(s)$ for loading, then the vehicle leaves $s$ to deliver all loaded freight to a subset of customers in $\mathcal{C}^D_s$.

The unload and load time windows at each supply point are defined by two parameters: the length of each unload and load time window (denoted by $len_u$ and $len_l$, respectively; we set $len_u = len_l$ in our experiment), and the difference between $e^l_s$ and $l^u_s$ (see Figure 9). We performed three runs with values of these two parameters equal to (20, 60), (30, 40) and (40,20) (remember that the length of time window at each supply point equals to 100 in all instances).

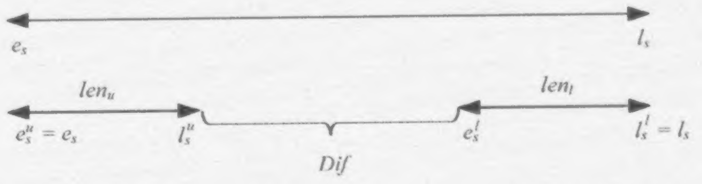


Figure 9: Illustration of two time windows at a supply point $s$

Table 8: Impact of synchronization at supply points on solution quality

| | One time window | Two time windows | | |
|---|---|---|---|---|
| | Len = 100, Dif = 0 | Len = 20, Dif = 60 | Len = 30, Dif = 40 | Len = 40, Dif = 20 |
| #Vehicles (%) | 0 | 1.03 | 0.70 | 0.52 |
| Traveling cost (%) | 0 | 2.17 | 0.82 | 0.74 |
| Total cost (%) | 0 | 1.94 | 0.88 | 0.75 |
| PD (%) | 49.88 | 45.40 | 46.98 | 48.31 |
| DM (%) | 26.94 | 22.45 | 23.95 | 24.01 |

The experiment was run on all instances. Table 8 sums up the solution-quality variations for three cases of two time windows compared to the case of only one time window.

The table displays the solution-quality variations in terms of the number of vehicles, traveling cost, and total cost. The percentage of times vehicles do both unload and load at supply points (*PD(%)* row) and the percentage of times vehicles move directly to a supply points without using waiting stations (*DM(%)* row) are also given.

Results indicate that solutions to the cases of two time windows are worse than those of one time window in terms of both number of vehicles and traveling cost. Moreover, longer waiting-time capabilities of supply points result in vehicles moving directly to supply points more frequently and doing more 'unload and load' operations (maximum of 26.94% and 49.88% respectively, both from the case of only one time window).

## 6.6 Comparing with the published results for the VRPB

As mentioned previously, the proposed algorithm can be directly applied to solve the VRPB which is a special case of the MZT-PDTWS. Hence, in this subsection, we compare our algorithm with the existing algorithms in the literature for the VRPB, in both cases with and without time windows.

### 6.6.1 Vehicle Routing problem with Backhauls and Time windows

In the Vehicle Routing problem with Backhauls and Time windows (VRPBTW), time windows at customers and duration of the route are considered. We have run our tabu search using only our routing neighborhoods on the Gélinas et al. (1995) 15 VRPBTW 100-customer instances. All parameters related to the supply points were discarded.

Different exact and meta-heuristic algorithms for the VRPBTW may be found in the literature. Gélinas et al. (1995) proposed a branching strategy for branch-and-bound approaches based on column generation. This algorithm found optimal solutions to 6 test problems. Potvin et al. (1996) designed a genetic algorithm, while the simple construction and improvement algorithms were developed by Thangiah et al. (1996). The ant system approach was used by Reimann et al. (2002) where only global pheromone updating was applied. A two-phase heuristic was proposed by Zhong and Cole (2005) in which customers were clustered in the first phase, then in the second phase, three route improvement routines (2-opt, 1-move, 1-exchange) running within a guided local search framework. Ropke and Pisinger (2006) developed a large neighborhood search which applied several competing removal and insertion heuristics. The selection of a heuristic was based on statistics gathered during the search.

Table 9 compares the performances obtained by our algorithm with the results of these algorithms. The first column gives the name of the authors of the study. Next fifteen columns present the number of vehicles and total distance with respect to 15 instances, respectively. These 15 instances are divided into five groups, i.e., R101, R102, R103, R104, R105, with three different percentages of backhaul customers (%BH) in each group. Finally, the rightmost column indicates the cumulative number of vehicles (CNV) and cumulative total distance (CTD) over all 15 instances. Most of algorithms in the literature (except Gélinas et al. (1995)) actually aimed first to reduce the number of vehicles. We do not, as our algorithm treats vehicles through supply point neighborhoods not considered in this experiments, and we do not compete with the other meta-heuristics on this count. We do compete with respect to the total distance, though, outperforming four out of the five meta-heuristics (with an average gap of 1.08%, a maximal gap of 2.81% and a minimal gap of -0.34%).

### 6.6.2 Vehicle Routing problem with Backhauls

The next round of experiments focused on the VRPB which is obtained by removing the constraints of time windows at customers and the route duration from the VRPBTW. The performance of the proposed tabu search is evaluated through comparison with results of other tabu search algorithms on two sets of instances in the VRPB literature. The first set of 62 instances was proposed in Goetschalckx and Jacobs-Blecha (1989). The instances range in size between 25 and 150 customers with backhauls ranging between 20 and 50%. The second set of 33 instances was proposed by Toth and Vigo (1997) with the number of customers range between 21 and 100, and backhauls percentages are either 20, 34 or 50%. In the VRPB literature, there are two different ways to compute the Euclidean distances between pairs of customers, namely real-valued and integer-valued cost matrix, respectively. The former matrix was used for all three tabu search algorithms with which we compare our method. Therefore, in this experiment, we only use real-valued cost matrix whose entries are the Euclidean distances.

Table 10 sums up the comparison of average of the best solutions for two sets of instances. The first column gives the name of the authors of the study. The average of the best solutions obtained by each study is given in the columns *Cost*. For completion sake, we also included the GAP for these studies relative to the average of best known solutions in the *GAP to BKS (%)* columns. One observes that the proposed TS performs well, outperforming all three other tabu search algorithms on Goetschalckx and Jacobs-Blecha (1989) instances (with an average gap of 0.05% and a maximal gap of 0.1%), and only worse than Brandão (2006) on Toth and Vigo (1997) instances (with a gap of -0.47%).

Table 9: Performance comparison with algorithms for the VRPBTW

| Authors | R101 | | | R102 | | | R103 | | | R104 | | | R105 | | | CNV/CTD |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | %BH | | | %BH | | | %BH | | | %BH | | | %BH | | | |
| | 10% | 30% | 50% | 10% | 30% | 50% | 10% | 30% | 50% | 10% | 30% | 50% | 10% | 30% | 50% | |
| Gélinas et al. (1995) | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| | 1767.9 | 1877.6 | 1895.1 | 1600.5 | 1639.2 | 1721.3 | - | - | - | - | - | - | - | - | - | - |
| Thangiah et al. (1996) | 24 | 24 | 25 | 20 | 21 | 21 | 15 | 16 | 17 | 13 | 12 | 13 | 17 | 18 | 18 | 274 |
| | 1842.3 | 1928.6 | 1937.6 | 1654.1 | 1764.3 | 1745.7 | 1371.6 | 1477.6 | 1543.2 | 1220.3 | 1302.5 | 1346.6 | 1553.4 | 1706.7 | 1657.4 | 24051.9 |
| Potvin et al. (1996) | 23 | 23 | 24 | 20 | 20 | 21 | 16 | 15 | 17 | 12 | 12 | 13 | 17 | 16 | 18 | 267 |
| | 1815 | 1896.6 | 1905.9 | 1622.9 | 1688.1 | 1735.7 | 1343.7 | 1381.6 | 1456.6 | 1117.7 | 1169.1 | 1203.7 | 1621 | 1652.8 | 1706.7 | 23317.1 |
| Reimann et al. (2002) | 22 | 23 | 24 | 19 | 22 | 22 | 16 | 15 | 17 | 11 | 12 | 12 | 16 | 16 | 17 | 265 |
| | 1831.68 | 1999.16 | 1945.29 | 1677.62 | 1754.43 | 1782.21 | 1348.41 | 1395.88 | 1467.66 | 1205.78 | 1128.3 | 1208.46 | 1544.81 | 1592.23 | 1633.01 | 23514.93 |
| Zhong and Cole (2005) | 24 | 24 | 25 | - | - | - | - | - | - | - | - | - | 17 | 17 | 19 | - |
| | 1848.04 | 2034.61 | 2057.05 | - | - | - | - | - | - | - | - | - | 1590.54 | 1667.92 | 1699.88 | - |
| Reimann and Ulrich (2006) | 22 | 23 | 24 | 19 | 22 | 22 | 15 | 15 | 17 | 11 | 11 | 11 | 16 | 16 | 17 | 261 |
| | 1853.45 | 1985.23 | 1964.04 | 1663.16 | 1759.02 | 1782.91 | 1454.25 | 1407.29 | 1478.48 | 1153.06 | 1228.62 | 1306.97 | 1570.11 | 1646.11 | 1689.74 | 23942.44 |
| Ropke and Pisinger (2006) | 22 | 23 | 24 | 19 | 22 | 22 | 15 | 15 | 17 | 11 | 11 | 11 | 15 | 16 | 16 | 259 |
| | 1818.86 | 1959.56 | 1939.1 | 1653.19 | 1750.7 | 1775.76 | 1387.57 | 1390.33 | 1456.58 | 1084.17 | 1154.84 | 1191.38 | 1561.28 | 1583.3 | 1710.19 | 23416.81 |
| TS | 22 | 23 | 24 | 19 | 22 | 22 | 15 | 15 | 17 | 11 | 11 | 12 | 16 | 16 | 17 | 263 |
| | 1823.64 | 1897.79 | 1917.87 | 1665.93 | 1758.31 | 1781.46 | 1402.63 | 1382.08 | 1471.43 | 1102.21 | 1181.17 | 1204.59 | 1571.42 | 1586.66 | 1648.32 | 23395.51 |

Table 10: Performance comparison with tabu search algorithms for the VRPB

| Authors | Goetschalckx and Jacobs-Blecha (1989) | | Toth and Vigo (1997) | |
|---|---|---|---|---|
| | Cost | GAP to BKS (%) | Cost | GAP to BKS (%) |
| Osman and Wassan (2002) | 291261.7 | 0.25 | 708.42 | 1.09 |
| Brandão (2006) | 291160.5 | 0.21 | 702.15 | 0.19 |
| Wassan (2007) | 290981.8 | 0.15 | 706.48 | 0.81 |
| TS | 290964.7 | 0.14 | 705.49 | 0.67 |

## 7 Conclusion

We studied the MZT-PDTWS, a new vehicle routing problem variant in which each vehicle performs multiple sequences of delivery and pickup through supply points within time synchronization restrictions. We proposed the first model formulation and a tabu search meta-heuristic integrating multiple neighborhoods for the problem. The computational study were performed on the first benchmark instances with up to 72 supply points and 7200 customer demands. Our experiments reveal that longer waiting-time capabilities of supply points tend to yield better results as it reduces the utilization of waiting stations, but also the number of empty trips. The MZT-PDTWS is a new problem and no previous results are available, we thus evaluated the performance of the proposed method through comparisons with published results on the VRPB as the MZT-PDTWS generalizes this problem. The experiments indicated that the proposed method is competitive with other meta-heuristics for both the cases with and without time windows.

## Acknowledgments

Partial funding for this project has been provided by the Natural Sciences and Engineering Council of Canada (NSERC), through its Industrial Research Chair and Discovery Grant programs, by the Université de Montréal, and by the EMME/2-STAN Royalty Research Funds (Dr. Heinz Spiess). We also gratefully acknowledge the support of Fonds de recherche du Québec through their infrastructure grants and of Calcul Québec and Compute Canada through access to their high-performance computing infrastructure.

## References

G. Berbeglia, J.-F. Cordeau, I. Gribkovskaia, and G. Laporte. Static pickup and delivery problems: a classification scheme and survey. *TOP*, 15:1–31, 2007.

G. Berbeglia, J.-F. Cordeau, and G. Laporte. Dynamic pickup and delivery problems. *European Journal of Operational Research*, 202:8–15, 2010.

J. Brandão. A new tabu search algorithm for the Vehicle Routing Problem with backhauls. *European Journal of Operational Research*, 173(2):540–555, 2006.

J.-F. Cordeau, G. Laporte, and A. Mercier. A unified Tabu Search heuristic for vehicle routing problems with time windows. *Journal of the Operational Research Society*, 52: 928–936, 2001.

T. G. Crainic, N. Ricciardi, and G. Storchi. Models for Evaluating and Planning City Logistics Systems. *Transportation Science*, 43(4):432–454, 2009.

T. G. Crainic, F. Errico, W. Rei, and N. Ricciardi. Integrating c2e and c2c Traffic into City Logistics Planning. *Procedia - Social and Behavioral Sciences*, 39(0):47–60, 2012a.

T. G. Crainic, Y. Gajpal, and M. Gendreau. Multi-Zone Multi-Trip Vehicle Routing Problem with Time Windows. Technical report, Publication CIRRELT-2012-36, Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport, Université de Montréal, Montréal, QC, Canada, 2012b.

M. Dell'Amico, G. Righini, and M. Salani. A Branch-and-Price Approach to the Vehicle Routing Problem with Simultaneous Distribution and Collection. *Transportation Science*, 40(2):235–247, 2006.

J. Dethloff. Relation between Vehicle Routing Problems: An Insertion Heuristic for the Vehicle Routing Problem with Simultaneous Delivery and Pick-Up Applied to the Vehicle Routing Problem with Backhauls. *The Journal of the Operational Research Society*, 53(1):115–118, 2002.

S. Gélinas, M. Desrochers, J. Desrosiers, and M. Solomon. A new branching strategy for time constrained routing problems with application to backhauling. *Annals of Operations Research*, 61(1):91–109, 1995.

M. Goetschalckx and C. Jacobs-Blecha. The Vehicle Routing Problem with backhauls. *European Journal of Operational Research*, 42(1):39–51, 1989.

I. Gribkovskaia, O. Halskau, and K. Myklebost. Models for Pick-up and Deliveries from Depots with Lasso Solutions. In *Proceedings of the 13th Annual Conference on Logistics Research*, pages 279–293. NOFOMA 2001, Collaboration in logistics: Connecting Islands using Information Technology, 2001.

Y. H. Lee, J. W. Jung, and K. M. Lee. Vehicle routing scheduling for cross-docking in the supply chain. *Computers & Industrial Engineering*, 51(2):247–256, 2006.

C.-J. Liao, Y. Lin, and S. C. Shih. Vehicle routing with cross-docking in the supply chain. *Expert Systems with Applications*, 37(10):6868–6873, 2010.

S. Lin. Computer solutions of the traveling salesman problem. *Bell System Technical Journal*, 44:2245–2269, 1965.

G. Nagy and S. Salhi. Heuristic algorithms for single and multiple depot vehicle routing problems with pickups and deliveries. *European Journal of Operational Research*, 162 (1):126–141, 2005.

P. K. Nguyen, T. G. Crainic, and M. Toulouse. A tabu search for Time-dependent Multi-zone Multi-trip Vehicle Routing Problem with Time Windows. *European Journal of Operational Research*, 231(1):43–56, 2013.

I. Or. *Traveling Salesman-type Combinatorial Problems and their relation to the Logistics of Blood Banking*. PhD thesis, Department of Industrial Engineering and Management Science, Northwestern University, Evanston, IL, 1976.

I. H. Osman. Meta strategy simulated annealing and tabu search algorithms for the Vehicle Routing Problem. *Annals of Operations Research*, 41:421–452, 1993.

I. H. Osman and N. Wassan. A reactive tabu search meta-heuristic for the vehicle routing problem with back-hauls. *Journal of Scheduling*, 5(4):263–285, 2002.

S. Parragh, K. Doerner, and R. Hartl. A survey on pickup and delivery problems. Part I: Transportation between customers and depot. *Journal für Betriebswirtschaft*, 58: 21–51, 2008a.

S. Parragh, K. Doerner, and R. Hartl. A survey on pickup and delivery problems. Part II: Transportation between pickup and delivery locations. *Journal für Betriebswirtschaft*, 58:81–117, 2008b.

J.-Y. Potvin, C. Duhamel, and F. Guertin. A genetic algorithm for vehicle routing with backhauling. *Applied Intelligence*, 6(4):345–355, 1996.

M. Reimann and H. Ulrich. Comparing backhauling strategies in vehicle routing using Ant Colony Optimization. *Central European Journal of Operations Research*, 14(2): 105–123, 2006.

M. Reimann, K. Doerner, and R. Hartl. Insertion Based Ants for Vehicle Routing Problems with Backhauls and Time Windows. In M. Dorigo, G. Caro, and M. Sampels, editors, *Ant Algorithms*, volume 2463 of *Lecture Notes in Computer Science*, pages 135–148. Springer Berlin Heidelberg, 2002.

S. Ropke and D. Pisinger. A unified heuristic for a large class of vehicle routing problems with backhauls. *European Journal of Operational Research*, 2004:750–775, 2006.

S. Salhi and G. Nagy. A Cluster Insertion Heuristic for Single and Multiple Depot Vehicle Routing Problems with Backhauling. *The Journal of the Operational Research Society*, 50(10):1034–1042, 1999.

M. W. P. Savelsbergh and M. M. Solomon. The general pickup and delivery problem. *Transportation Science*, 29:17–29, 1995.

M. M. Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35:254–265, 1987.

E. D. Taillard, P. Badeau, M. Gendreau, F. Guertin, and J.-Y. Potvin. A tabu search heuristic for the Vehicle Routing Problem with soft time windows. *Transportation Science*, 31:170–186, 1997.

S. R. Thangiah, J.-Y. Potvin, and T. Sun. Heuristic approaches to vehicle routing with backhauls and time windows. *Computers & Operations Research*, 23(11):1043–1057, 1996.

P. Toth and D. Vigo. An Exact Algorithm for the Vehicle Routing Problem with Backhauls. *Transportation Science*, 31(4):372–385, 1997.

P. Toth and D. Vigo. *The Vehicle Routing Problem*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2002.

N. Wassan. Reactive Tabu Adaptive Memory Programming Search for the Vehicle Routing Problem with Backhauls. *The Journal of the Operational Research Society*, 58(12): 1630–1641, 2007.

M. Wen, J. Larsen, J. Clausen, J.-F. Cordeau, and G. Laporte. Vehicle Routing with Cross-Docking, 2008.

Y. Zhong and M. H. Cole. A vehicle routing problem with backhauls and time windows: a guided local search solution. *Transportation Research Part E: Logistics and Transportation Review*, 41(2):131–144, 2005.

## Annex A. Detailed Results

Tables A11, A12, A13, A14, A15, and A16 display the detailed results obtained by the proposed tabu search, the average values (*Avg10* column), standard deviations (*Std* column), and the best solution values (*Best10* column) over 10 runs, the number of vehicles (*#Vehicles* column), the number of times vehicles move directly to a supply point without passing through waiting stations (*DM* column), the number of times waiting stations are used for moving between customer zones (*MWS* column), the number of times vehicles do both 'unload and load' operation once they arrive at supply points (*PD* column), the number of legs (*#Legs* column).

Table A11: Detailed results on problem instances set A1

| Instance | Avg10 | Std | Best10 | #Vehicles | DM | MWS | PD | #Legs |
|---|---|---|---|---|---|---|---|---|
| A1-1 | 19125.65 | 35.52 | 19052.70 | 18 | 3 | 42 | 8 | 70 |
| A1-2 | 20627.00 | 41.79 | 20532.30 | 24 | 2 | 35 | 9 | 68 |
| A1-3 | 17555.37 | 77.06 | 17438.78 | 18 | 0 | 42 | 9 | 68 |
| A1-4 | 24232.20 | 95.98 | 24027.82 | 31 | 0 | 31 | 9 | 69 |
| A1-5 | 17826.23 | 44.93 | 17741.77 | 18 | 16 | 30 | 9 | 72 |
| A1-6 | 19768.53 | 40.51 | 19694.70 | 18 | 17 | 33 | 27 | 89 |
| A1-7 | 21709.69 | 67.39 | 21572.62 | 24 | 9 | 30 | 25 | 85 |
| A1-8 | 18536.57 | 87.90 | 18292.37 | 18 | 13 | 33 | 26 | 85 |
| A1-9 | 25565.77 | 100.09 | 25382.80 | 31 | 1 | 32 | 26 | 87 |
| A1-10 | 19457.42 | 92.58 | 19328.76 | 19 | 19 | 28 | 27 | 90 |
| A1-11 | 21572.33 | 79.16 | 21399.10 | 18 | 43 | 19 | 57 | 123 |
| A1-12 | 24223.69 | 93.21 | 23978.20 | 24 | 28 | 31 | 46 | 119 |
| A1-13 | 20797.80 | 70.05 | 20652.90 | 18 | 31 | 32 | 55 | 118 |
| A1-14 | 28375.16 | 112.27 | 28136.80 | 31 | 18 | 40 | 55 | 118 |
| A1-15 | 22390.35 | 139.26 | 22061.10 | 20 | 39 | 29 | 57 | 125 |
| Average | 21450.92 | 78.51 | 21286.18 | 22.00 | 15.93 | 32.47 | 29.67 | 92.40 |

Table A12: Detailed results on problem instances set A2

| Instance | Avg10 | Std | Best10 | #Vehicles | DM | MWS | PD | #Legs |
|---|---|---|---|---|---|---|---|---|
| A2-1 | 18577.68 | 98.24 | 18410.01 | 19 | 6 | 40 | 12 | 76 |
| A2-2 | 19725.88 | 87.47 | 19590.37 | 21 | 12 | 30 | 8 | 70 |
| A2-3 | 15885.33 | 87.89 | 15767.75 | 16 | 9 | 40 | 10 | 74 |
| A2-4 | 14923.72 | 42.97 | 14859.18 | 13 | 1 | 49 | 11 | 72 |
| A2-5 | 15307.64 | 91.77 | 15156.95 | 13 | 7 | 43 | 10 | 72 |
| A2-6 | 20286.66 | 97.82 | 20113.70 | 19 | 16 | 32 | 30 | 95 |
| A2-7 | 21459.12 | 99.36 | 21321.10 | 21 | 21 | 25 | 29 | 92 |
| A2-8 | 17541.52 | 97.37 | 17339.61 | 16 | 17 | 35 | 28 | 91 |
| A2-9 | 16378.68 | 80.03 | 16209.82 | 13 | 6 | 47 | 28 | 89 |
| A2-10 | 16646.85 | 88.86 | 16494.59 | 13 | 20 | 33 | 28 | 90 |
| A2-11 | 22422.22 | 91.86 | 22195.60 | 20 | 24 | 41 | 55 | 127 |
| A2-12 | 23920.24 | 56.28 | 23823.10 | 21 | 38 | 27 | 55 | 124 |
| A2-13 | 21312.70 | 97.21 | 21079.20 | 18 | 23 | 39 | 58 | 126 |
| A2-14 | 19055.70 | 90.23 | 18901.30 | 14 | 22 | 40 | 57 | 121 |
| A2-15 | 19043.00 | 84.86 | 18906.10 | 13 | 37 | 27 | 55 | 124 |
| Average | 18832.46 | 86.15 | 18677.89 | 16.67 | 17.27 | 36.53 | 31.60 | 96.20 |

Table A13: Detailed results on problem instances set B1

| Instance | Avg10 | Std | Best10 | #Vehicles | DM | MWS | PD | #Legs |
|---|---|---|---|---|---|---|---|---|
| B1-1 | 83561.38 | 35.75 | 83498.7 | 77 | 30 | 151 | 30 | 282 |
| B1-2 | 62675.24 | 70.27 | 62513.8 | 40 | 50 | 158 | 33 | 277 |
| B1-3 | 59566.89 | 103.07 | 59281.3 | 36 | 45 | 171 | 30 | 276 |
| B1-4 | 66674.47 | 119.30 | 66454.1 | 41 | 39 | 168 | 31 | 273 |
| B1-5 | 62420.99 | 139.67 | 62071.1 | 40 | 33 | 174 | 29 | 271 |
| B1-6 | 94815.97 | 105.30 | 94634.4 | 77 | 47 | 171 | 104 | 358 |
| B1-7 | 70117.71 | 123.55 | 69773.4 | 43 | 72 | 150 | 104 | 349 |
| B1-8 | 69328.12 | 113.24 | 69139.9 | 37 | 82 | 148 | 106 | 356 |
| B1-9 | 72630.66 | 74.17 | 72449.4 | 40 | 79 | 150 | 103 | 352 |
| B1-10 | 71042.78 | 32.74 | 70994 | 42 | 69 | 147 | 104 | 347 |
| B1-11 | 115479.33 | 105.64 | 115313.5 | 80 | 70 | 205 | 219 | 495 |
| B1-12 | 90142.91 | 85.73 | 90053.1 | 46 | 120 | 146 | 211 | 484 |
| B1-13 | 93478.07 | 73.38 | 93389.9 | 49 | 119 | 152 | 213 | 492 |
| B1-14 | 99444.83 | 72.84 | 99349.3 | 46 | 87 | 176 | 216 | 485 |
| B1-15 | 97233.69 | 97.95 | 97024 | 53 | 114 | 142 | 211 | 480 |
| Average | 80574.20 | 90.17 | 80395.99 | 49.80 | 70.40 | 160.60 | 116.27 | 371.80 |

Table A14: Detailed results on problem instances set B2

| Instance | Avg10 | Std | Best10 | #Vehicles | DM | MWS | PD | #Legs |
|---|---|---|---|---|---|---|---|---|
| B2-1 | 57606.41 | 97.85 | 57406.8 | 31 | 48 | 174 | 40 | 291 |
| B2-2 | 64177.54 | 68.39 | 64048 | 45 | 47 | 165 | 36 | 285 |
| B2-3 | 61249.92 | 90.45 | 61096.2 | 36 | 43 | 182 | 38 | 292 |
| B2-4 | 58321.89 | 108.80 | 58286.9 | 32 | 35 | 194 | 33 | 289 |
| B2-5 | 57787.64 | 77.15 | 57749.5 | 38 | 38 | 181 | 36 | 288 |
| B2-6 | 69084.69 | 68.22 | 68991.4 | 38 | 45 | 178 | 104 | 353 |
| B2-7 | 77033.95 | 85.71 | 76810.3 | 48 | 68 | 169 | 112 | 366 |
| B2-8 | 73010.21 | 70.96 | 72821.9 | 36 | 59 | 174 | 106 | 360 |
| B2-9 | 70535.79 | 58.06 | 70448.2 | 36 | 52 | 174 | 103 | 353 |
| B2-10 | 70829.02 | 96.03 | 70656 | 42 | 46 | 177 | 109 | 361 |
| B2-11 | 91315.56 | 98.60 | 91199.3 | 41 | 86 | 180 | 211 | 487 |
| B2-12 | 100857.73 | 60.85 | 100728.3 | 60 | 95 | 177 | 220 | 493 |
| B2-13 | 97624.36 | 102.62 | 97410.2 | 41 | 85 | 192 | 209 | 493 |
| B2-14 | 89699.15 | 131.32 | 89332.2 | 42 | 91 | 191 | 219 | 504 |
| B2-15 | 90624.95 | 47.31 | 90522.6 | 46 | 92 | 187 | 220 | 500 |
| Average | 75317.25 | 84.16 | 75167.19 | 40.80 | 62.00 | 179.67 | 119.73 | 381.00 |

Table A15: Detailed results on problem instances set C1

| Instance | Avg10 | Std | Best10 | #Vehicles | DM | MWS | PD | #Legs |
|---|---|---|---|---|---|---|---|---|
| C1-1 | 154322.30 | 72.45 | 154127 | 93 | 89 | 374 | 65 | 616 |
| C1-2 | 150028.70 | 142.39 | 149835 | 92 | 82 | 374 | 65 | 607 |
| C1-3 | 152287.90 | 135.71 | 152100 | 83 | 95 | 386 | 65 | 619 |
| C1-4 | 154935.90 | 57.31 | 154805 | 97 | 68 | 393 | 63 | 616 |
| C1-5 | 155101.20 | 161.33 | 154665 | 87 | 78 | 395 | 65 | 619 |
| C1-6 | 203099.00 | 120.09 | 202994 | 101 | 116 | 372 | 225 | 786 |
| C1-7 | 196721.20 | 165.82 | 196335 | 101 | 100 | 397 | 228 | 786 |
| C1-8 | 200641.20 | 33.40 | 200559 | 91 | 120 | 395 | 231 | 787 |
| C1-9 | 198211.80 | 110.56 | 197954 | 107 | 106 | 375 | 232 | 792 |
| C1-10 | 201688.80 | 162.98 | 201402 | 97 | 99 | 401 | 227 | 799 |
| C1-11 | 293262.00 | 153.85 | 293078 | 123 | 156 | 426 | 501 | 1106 |
| C1-12 | 285020.80 | 81.31 | 284801 | 121 | 203 | 391 | 484 | 1086 |
| C1-13 | 293493.00 | 155.59 | 293226 | 112 | 221 | 370 | 491 | 1100 |
| C1-14 | 286054.90 | 91.54 | 285829 | 119 | 162 | 423 | 497 | 1098 |
| C1-15 | 302333.50 | 48.99 | 302249 | 124 | 172 | 433 | 469 | 1112 |
| Average | 215146.81 | 112.89 | 214930.60 | 103.20 | 124.47 | 393.87 | 260.53 | 835.27 |

Table A16: Detailed results on problem instances set C2

| Instance | Avg10 | Std | Best10 | #Vehicles | DM | MWS | PD | #Legs |
|---|---|---|---|---|---|---|---|---|
| C2-1 | 142736.00 | 109.07 | 142511 | 77 | 114 | 391 | 78 | 655 |
| C2-2 | 141224.60 | 55.47 | 141095 | 77 | 97 | 414 | 80 | 663 |
| C2-3 | 147322.30 | 61.62 | 147219 | 82 | 86 | 426 | 80 | 657 |
| C2-4 | 134630.60 | 42.95 | 134567.2 | 73 | 83 | 416 | 79 | 642 |
| C2-5 | 139177.10 | 196.80 | 138623 | 72 | 83 | 426 | 80 | 655 |
| C2-6 | 188064.70 | 202.55 | 187501 | 89 | 144 | 392 | 232 | 817 |
| C2-7 | 224601.60 | 118.17 | 224486 | 103 | 222 | 408 | 236 | 900 |
| C2-8 | 191738.70 | 210.74 | 191220 | 98 | 113 | 433 | 232 | 817 |
| C2-9 | 183440.30 | 166.22 | 183034 | 92 | 100 | 428 | 233 | 812 |
| C2-10 | 190020.60 | 87.56 | 189789 | 90 | 106 | 440 | 235 | 833 |
| C2-11 | 290027.30 | 75.65 | 289826 | 117 | 198 | 451 | 484 | 1142 |
| C2-12 | 265985.50 | 210.68 | 265391 | 88 | 148 | 479 | 494 | 1132 |
| C2-13 | 276053.30 | 107.45 | 275884 | 122 | 177 | 455 | 477 | 1126 |
| C2-14 | 274952.90 | 137.77 | 274742 | 110 | 148 | 482 | 485 | 1132 |
| C2-15 | 284755.10 | 160.20 | 284448 | 97 | 167 | 463 | 524 | 1157 |
| Average | 204982.04 | 129.53 | 204689.08 | 92.47 | 132.40 | 433.60 | 268.60 | 876.00 |